STRUCTURED SAMPLING AND RECONSTRUCTION OF ILLUMINATION FOR IMAGE SYNTHESIS

by

Georgios Drettakis

A thesis submitted in conformity with the requirements for the
Degree of Doctor of Philosophy,
Graduate Department of Computer Science,
University of Toronto

A Laurence

Structured Sampling and Reconstruction of Illumination for Image Synthesis

Doctor of Philosophy, 1994

Georgios Drettakis
Graduate Department of Computer Science, University of Toronto

Abstract

An important goal of image synthesis is to achieve accurate, efficient and consistent sampling and reconstruction of illumination varying over surfaces in an environment. A new approach is introduced for the treatment of diffuse polyhedral environments lit by area light sources, based on the identification of important properties of illumination structure. The properties of unimodality and curvature of illumination in unoccluded environments are used to develop a high quality sampling algorithm which includes error bounds. An efficient algorithm is presented to partition the scene polygons into a mesh of cells, in which the visible part of the source has the same topology. A fast incremental algorithm is presented to calculate the backprojection, which is an abstract representation of this topology. The behaviour of illumination in the penumbral regions is carefully studied, and is shown to be monotonic and well behaved within most of the mesh cells. An algorithm to reduce the mesh size, and an algorithm which selects between linear and quadratic interpolants are presented. The results show that the mesh size and the degrees of the interpolants can be reduced without significant degradation of image quality. The preceding algorithms are combined into a complete structured sampling approach that allows accurate and efficient representation of illumination using interpolating polynomials for scenes with occlusion. Images with accurate shadows can be produced from the structured representation using either ray-casting or polygon rendering hardware. Finally, it is shown that our methodology generalises easily to the global illumination problem. An iterative solution to a Galerkin finite element approach is proposed, and it is shown how the structured algorithms provide a good initial approximation for the iteration, enhance efficiency for numerical integration and allow adaptive mesh modification. The structure-driven global illumination algorithm thus promises significant improvement over previous higher-order finite element solutions.

Acknowledgments

Eugene Fiume has been the best supervisor a Ph.D. student could hope for. He has been a great friend over these past years, and has always been supportive, encouraging and inspiring. Despite an enormously busy professional and family life, he always found the time to provide invaluable guidance throughout the long and sometimes painful thesis process. Many thanks Eugene!

Thanks are due to the members of my committee, Christina Christara, Allan Jepson, James Stewart and Demetri Terzopoulos who took the time to read the thesis carefully and offered important input and observations throughout the Ph.D. process. Thanks also to John Curran for serving on my committee and to Don Fussell, my external appraiser, who read the thesis very thoroughly and made very insightful suggestions. Thanks to Marc Ouellette who read drafts of some chapters.

Many thanks to the University of Toronto, NSERC, ITRC and Mrs. Monica Ryckman who provided financial support for my studies, and the Dynamic Graphics Project (DGP) lab for offering me part-time employment as a sys-admin and later the lab manager.

The implementation of the research in this thesis is a C++ monster of 75 thousands lines of code. About 30% of that was contributed in C (and subsequently converted to C++ by yours truly) by the following people: Rob Lansdale, who wrote the original input parsing and entire rendering back-end system, John Amanatides, Pierre Poulin and Andrew Woo who wrote the various ray-tracing elements, James Stewart who wrote a first implementation of the extended winged-edge data structure, and also Jos Stam and Marc Ouellette who contributed other pieces of code.

Victor Ng, Dimitris Nastos, Alex Mitchell and Kevin Schlueter helped keep the systems running, especially during the last few months while I was in the deep end. It was a pleasure working with them to keep DGP running smoothly. Tom Milligan and Garry Beirne have often helped out in many ways.

The love and patience of my fiancee Laurence Bianchini, to whom this thesis is dedicated, has kept me strong and my spirits high for the past two years. I cannot envision having completed this thesis without her. Many thanks also to my parents, Manolis and Maria, and to my sister Artemis, for their love and support which I can always count on.

Since the Ph.D. is the end of the formal educational process it is only appropriate to mention some of the people who influenced my decisions in post-secondary education. Dionysis Tsichritzis was the one who convinced me to change to Computer Science from Mathematics; Dennis Allison, Nikos Spyratos, Steve Hull and Rob Pike each in his own particular way showed me that CS could be a lot of fun and did not necessarily mean that you have to be a geek. Manolis Katevenis, Panos Constantopoulos, Christos Nikolau and Yannis Vassiliou were excellent teachers in my undergrad years at the University of Crete.

DGP is the best place I can think of to study, and do research in, Computer Graphics. The quality of research is outstanding and our group always seems to have a steady flow of intelligent, fun people, who know when to do great research and when to enjoy themselves. I feel that I got a solid education here, and learned to love learning again. My contemporary Ph.D. students all often put up with my rantings about my research and complaints about the weather: Mike van de Panne with his endless source of energy, Jos Stam always there to explain the math or go for a beer, Michael McCool also a good source of math expertise and references and Marc Ouellette, the only other illumination guy, whose bad jokes have gotten me through many long nights of work. They have all contributed to this work in many ways as well as being good friends. Pierre Poulin, once a DGP'er, as well as Gavin Miller and Eric Chen from Apple have also served as sounding boards for ideas and provided inspiration for research at various stages of this work.

Dimitris Nastos, Dimitra and Spiros have been good friends; it has been a rare privilege to be able to speak Greek on an everyday basis in such a far off place. Many nights of beers and wings, many movies and general social events with Beverly, Tim, George F., Tom and Moira, Maria and earlier Ian, Gord and Blaine, as well as all the others from the lab and the department, have kept me out of trouble all these years. The "new greek generation", Dimitris A., Michalis, Petros and Vassilis kept me company and helped me get through the tough last stages of this thesis. My patient roommate Mike (MC), who always remembered to take out the garbage, has been a good friend and made sharing a place to live a pleasure rather than a burden.

Life in Canada and in particular the city of Toronto has been a great experience. The unique mix of cultures and the opportunity to meet people from so many different places has broadened my outlook and made me a much more tolerant and open-minded person. It has been a great stay here in TO. Thanks to you all, eh!

Table of Contents

List of Figures

List of Tables

List of Colour Images

Chapter 1

Introduction

1.1 Motivation

The creation of realistic images has been one of the major goals of computer graphics research. An important aspect of this problem is the calculation of light transport in such a way that visual effects such as shadows and indirect illumination are correctly represented in the synthetic images. The quantity commonly calculated for this representation is *radiance*, that is light power per area per solid angle.

Given a scene S consisting of a set of objects, a set of light sources E, and a set of reflectivity functions f_r associated with each surface, we wish to calculate the radiance reflected from each surface. The computed radiance is then used to synthesize realistic lighting for computer generated images. The problem of calculating radiance at a point in a scene is difficult. We must account for the light received directly from the light sources, as well as the light reflected off one surface, arriving at another surface and so on until the light power transport is no longer significant. Among the factors that complicate this process are shadows, which are caused when an object blocks the light path from an emitter to a receiver, and surfaces with complicated reflection functions such as mirrors or anisotropic materials.

Formally, the radiance at a given point $\mathbf{x}'$ in the environment is given by an integral equation. This integral equation characterises all light transfers, from sources and surfaces in the environment. This transport is described in terms of the radiance at point $\mathbf{x}'$ on (a non-emitting) surface *s*, in the direction of point $\mathbf{x}''$ ([37][30]), due to other points $\mathbf{x}$ in the environment:

$$L(\mathbf{x}', \mathbf{x}'') = \int_{\Omega} f_r(\mathbf{x} \to \mathbf{x}' \to \mathbf{x}'')\, L(\mathbf{x}, \mathbf{x}')\, G(\mathbf{x}, \mathbf{x}')\, d\mathbf{x}, \tag{1-1}$$

where f_r is the reflection function for any point $\mathbf{x}$ on a surface *s,* reflected at $\mathbf{x}'$ in the direction of $\mathbf{x}''$, Ω is the hemisphere above point $\mathbf{x}'$, and $G(\mathbf{x}, \mathbf{x}')$ is a geometric function which is 0 if $\mathbf{x}'$ is not visible from $\mathbf{x}$, and depends on the relative geometric configuration of $\mathbf{x}'$ and $\mathbf{x}$ everywhere else. To calculate global illumination for image synthesis it is necessary to approximate the solution of this integral equation.

Eq. (1-1) rarely has a closed form solution for the functions $L(\mathbf{x}, \mathbf{x}')$ of radiance over every surface, even for simple environments, due to the interdependence of the radiance of one surface on the radiance of every other and the geometric complexity implied by $G(\mathbf{x}, \mathbf{x}')$. In previous work, researchers have attempted to find approximate solutions to the equations.

Early attempts used a regular grid to discretise the surfaces in the environment, and assumed constant radiance over grid elements, reducing Eq. (1-1) to a simpler linear system ([29]). Various solution techniques for this simpler system have been developed, most of them providing approximate discrete solutions to the visibility variation between grid elements (e.g. hemi-cube [14] or ray-tracing [64]), coupled with adaptive subdivision strategies. The subdivision criteria vary (radiance gradient [15], power transfer [30], view dependence [56]), but all depend on the grid discretisation. To further improve the solutions to the simplified system, linear or higher degree approximations over grid elements have been used ([72][64][27]). Some first attempts have recently been made to resolve the geometric part of the problem by directly computing shadow boundaries ([9][34][39][38]). Realistic images have been created using many of these methods, often for quite complex and interesting environments.

In previous methods it has been assumed, to varying degrees, that the function of radiance across a surface is unknown. This is natural since in most cases the solution to the global light transport problem expressed in (1-1) is a complicated function, and thus its behaviour can be difficult to predict. This difficulty, and the unknown character of the solution, has made error analysis of the approximate solutions computed by these algorithms problematic. These issues underline the shortcomings of most previous research. By attempting to solve the complicated global problem the properties of radiance cannot be determined or understood. Consequently, error analysis of the approximate solutions found is difficult. Inevitably, these solutions will require increased computation and storage, since approximating an unknown function is significantly more challenging than approximating a function with known behaviour.

In this thesis we propose an alternative methodology to address these issues. The research of this thesis starts with the study of radiance in simple environments, specifically the study of illumination received directly from constant diffuse emitters in environments with diffuse reflectors. Despite the fact that even in these environments the analytic and geometric properties of radiance can be difficult to compute, it is possible to determine several important properties of the radiance function. These properties convey the *structure* of illumination.

The analytic and geometric properties of radiance are crucial to the understanding of the phenomena that are being simulated. It is possible to develop analytic solutions that can be computed efficiently for interesting environments of this kind, thus facilitating careful error analysis and evaluation. The understanding achieved through the recognition of these properties, and the ability to carefully analyse the error, leads to the development of more efficient and accurate sampling and reconstruction algorithms. Finally, the structured methodology is not restricted to simple environments; these first steps serve as a sound basis for the approximations that will be subsequently applied to the general global illumination problem.

1.2 Contributions

Following the general principles outlined above, the research of this thesis has resulted in four general contributions.

1. The behaviour, or *structure*, of radiance has been studied and characterised both for unoccluded environments, and for more complex environments involving occlusion. Formal and empirical methods have been used to show that unoccluded radiance from an interesting class of light sources displays a single maximum and varies slowly away from that maximum. In the presence of partial occlusion, radiance has been empirically studied, and determined to be mostly monotonic within regions for which the topological structure of the visible region of the

source is the same. The most common cases for which extrema occur in the penumbra have also been identified and characterised. The geometric structure of visibility directly affecting radiance has also been characterised.

2. An efficient boundary calculation algorithm has been developed. This algorithm segments the surfaces in an environment into regions, or cells, within which the topological structure of the visible region of an area light source is the same. The geometric structure of visibility is used to develop heuristics that result in good observed running time with respect to the number of objects in the scene. An incremental algorithm is also introduced that allows the calculation of an abstract representation of the visible portion of the source, called a backprojection, in each mesh cell at minimal additional cost. This approach allows efficient computation of images for which exact radiance values are calculated at every pixel, including those in the penumbra, while the backprojection is central in the study of radiance in penumbral regions.

3. A sparse and accurate piecewise polynomial approximation of radiance is built for scenes with occlusion, based on illumination structure. The shadow mesh is used to specify the domains over which the interpolants are defined, while radiance structure in unoccluded and penumbral regions is used to develop a sparse sampling and reconstruction algorithm. Using radiance structure properties, tight error bounds are built into these approximations. In the penumbral regions, both the interpolant degree and the geometric complexity of the mesh are reduced again based on radiance properties, which permits high quality interpolation to be maintained.

4. The construction of the piecewise polynomial interpolant of radiance over the surfaces of the environment has been used to formulate a general higher-order Galerkin-type finite element method for global illumination transport. This formulation shows that the structured sampling algorithms developed for direct illumination provide significant advantages in a modified progressive refinement global illumination algorithm using higher order interpolants.

1.3 Results

The basic research results of this thesis are presented in Chapters 3-6. These results are now summarised.

Chapter 3: Structured Sampling for Unoccluded Illumination

From the study of unoccluded illumination it will be established that radiance due to diffuse disc light sources on a planar receiver of arbitrary orientation is unimodal. It is conjectured that radiance from convex polygonal light sources is also unimodal. Unimodality and curvature of radiance functions are identified as properties that are useful for the development of a structured sampling algorithm.

The first step of the structured sampling and reconstruction algorithm for unoccluded illumination is the determination of the maximum of radiance on the surface, if such a maximum exists. Subsequently radiance is approximated on two orthogonal axes passing through that maximum, and the inflection points are found along these axes if they exist. Adaptive subdivision is then performed along these lines, based on derivative analysis. Strict error bounds are guaranteed from the structure-driven subdivision process. The grid resulting from this subdivision is used to build a sparse two-dimensional interpolant of the radiance function on the receiving surfaces.

Tests from the implementation of the structured algorithm show the quality of the images generated with the same number of samples as a uniform solution is between 2 and 8 times better, depending on the scene and the error metric used. If additional samples are used to compensate for the expense of subdivision, the structured algorithm still performs better in terms of pixel-by-pixel image quality, and in addition displays superior subjective visual quality by reducing annoying artifacts such as banding. Finally, the structured algorithm displays consistency in animation, since the structured sampling strategy follows the movement of important radiance characteristics with the motion of the source.

Chapter 4: Computing Penumbral Boundaries

In an environment with occlusion, the complete discontinuity mesh for an area source is a segmentation of the polygons in a scene into cells or *faces*. In each such face the topological structure of the visible region of the source is the same. The geometry of the visible part of the source is represented in an abstract fashion by the *backprojection*. This representation allows the efficient determination of the visible part of a source at any point in a mesh face without the need to query the scene and perform visibility calculations.

A fast practical algorithm to compute the complete discontinuity mesh is developed next. Important properties of geometric structure are identified and statistics are run to support these observations on a set of typical scenes. These properties are then used to accelerate the mesh construction.

The mesh is computed by casting discontinuity surfaces that cut the scene polygons along lines or curves that delineate the faces of the mesh. The algorithm classifies the types of discontinuity surfaces depending on their frequency of occurrence, and takes advantage of scene structure to efficiently perform the identification of the important surfaces and to accelerate subsequent casting. We observe that many of these discontinuity surfaces cut a small number of polygons, and thus the number of corresponding curves to be inserted in the mesh is small, and also that the resulting mesh is such that incremental insertion can be performed efficiently. These observations are used to accelerate the mesh construction. The use of a voxel grid allows efficient queries of the scene. The implementation of the mesh construction algorithm was run on a set of scenes of office interiors. The running times observed display close to linear growth with respect to the number of objects due to the utilisation of geometric structure, despite pessimistic worst case complexity.

An incremental algorithm is presented that allows the computation of the backprojection information in each face of the mesh to be performed just by traversing edges, and thus does not require any visibility processing or scene query. The use of the backprojection allows the exact computation of radiance for images with occlusion to be performed efficiently for the first time. Also, the use of the incremental algorithm drastically reduces the cost of computing illumination that was equal or greater than the cost of mesh construction in previous approaches.

Chapter 5: Structured Sampling for Scenes with Occlusion

The properties of illumination in penumbral regions have been extensively studied. The computation of the complete mesh with backprojections facilitates the collection of statistics on radiance function behaviour. This study shows that in the vast majority of faces in the mesh, radiance is smooth and monotonic. The geometry of the backprojection, the relative position and orientation of the source with respect to the receiver and the interaction with unoccluded illumination are the dominating factors contributing to non-monotonic or highly variable behaviour of radiance in the remaining penumbral faces.

The effect of the derivative discontinuities delineated by mesh lines is discussed, and it is observed that for many cases the effect on the variation of the function is minimal. In addition, it is observed that radiance varies little in small faces. Using these two observations the mesh is reduced by eliminating small faces and faces that cause small concavities. In contrast, for the infrequent cases in which it is required, some faces are adaptively subdivided. The resulting mesh is used to create a triangulation of the penumbral domain, on which interpolants are built.

The edges of the triangulation are processed to select a suitable degree, linear or quadratic, depending on the variation of radiance along the edge in question. Lagrange triangular basis functions are then constructed, for which C^0 continuity is guaranteed. Error bound analysis is performed for the resulting interpolants.

The interpolant construction is combined with the structured sampling algorithm for the unoccluded regions, by segmenting the areas of light into regular regions, defined by the bounding boxes of groups of penumbral faces. The result is a complete structured sampling algorithm for environments with occlusion, in which radiance is represented using accurate, low-order piecewise polynomial interpolants. Rendering is performed by ray-casting or graphics workstation interpolation-based hardware.

The results from the implementation show that the structured approximation shows low error, and both the mesh reduction and degree selection algorithms perform well, respectively reducing mesh size and interpolant degree, without significant deterioration of image quality.

Chapter 6: A Structure-Driven Global Illumination Algorithm

A modified Galerkin-type progressive refinement approach is formulated for the solution of the global illumination problem, using the low-order polynomial representation built by the structured sampling algorithm.

The methods of the structured algorithm provide significant improvement over previous higher-order methods for global illumination computations. Specifically, the interpolants representing the radiance due to direct illumination created by the structured method are used as a starting point for the iterative solution. Also, the numerical computation of integrals required by the Galerkin method is made significantly more efficient by the segmentation into unoccluded regions and regions of partial illumination. For the regions of partial occlusion in which high precision is required, the backprojection allows the efficient numerical computation of the integrals with high accuracy.

Finally, the structured method iteratively modifies the meshes used in the solution process, and matches the quality of the solution to the required accuracy. The study of radiance structure facilitates the determination of heuristics that allow varying quality visibility computations to be performed, that are expected to lead to faster convergence.

1.4 Thesis Structure

In Chapter 2, previous work in the area of illumination is surveyed. This includes solution methods for light transport, sampling techniques for illumination reconstruction and some results from heat transfer and illumination engineering used extensively in global illumination research. A discussion of related work on shadow algorithms is also presented.

Following this survey of previous work, the research results of this thesis are presented in Chapters 3 through 6, as outlined in Section 1.3.

The thesis closes in Chapter 7, with a brief overview of the results and a discussion of the extensions of the structured approach and future directions.

Chapter 2

Previous Work

The focus of this thesis is on the sampling and reconstruction of radiance functions defined over receiver surfaces, on the geometry of shadows and the global illumination problem. It is therefore important to overview relevant previous work, mainly from the domain of computer graphics. Results from illumination engineering that lend important insight into the nature of illumination are also reviewed, as well as work in computer vision that is fundamental to the understanding of shadow geometry in polyhedral environments.

2.1 Calculating Illumination for Image Synthesis

In illumination for image synthesis, the quantity calculated is typically radiance (light power per area per solid angle). Radiance is used since it measures light power and is closely related to what the eye perceives. Radiance can also be used to describe general oriented light behaviours, due to its dependence on solid angle. For a differential emitting area dS of an emitting surface (light source or secondary reflector), the radiance reaching a point $P(\mathbf{x})$ at distance R (Figure 2-1) is

$$dL(dS, \mathbf{x}) = M\cos\theta_{out}d\omega_i = M\frac{\cos\theta_{out}\cos\theta_{in}}{R^2}dS, \tag{2-1}$$

where M is the luminosity of the light source at the point dS and $\cos\theta_{out}$ is the angle formed by the normal to the surface of the source and the line connecting P and dS. The solid angle of the cone that has P as its apex and dS as its base is $d\omega_i$, that is equal to $\cos\theta_{in}/R^2$, where $\cos\theta_{in}$ is the angle of the normal to the receiver at P to the line connecting it to dS (see Figure 2-1).

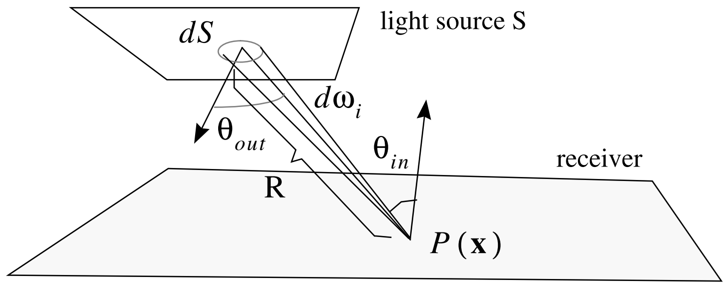


Figure 2-1. Basic Illumination Quantities

To account for the contribution from the whole source, the differential radiance is integrated over the entire area source S:

$$L_S(\mathbf{x}) = \int_S dL(dS, \mathbf{x}) = \int_S M \frac{\cos\theta_{out}\cos\theta_{in}}{R^2} dS. \tag{2-2}$$

Consider a visible point $\mathbf{x}'$, shown in Figure 2-2. To calculate the radiance that leaves $\mathbf{x}'$ and reaches the eye, positioned at $\mathbf{x}$, it is necessary to account for the radiance $\varepsilon(\mathbf{x}, \mathbf{x}')$ emitted from $\mathbf{x}'$ to $\mathbf{x}$, and the radiance impinging through the hemispherical solid angle Ω above $\mathbf{x}'$, that is then reflected in the direction of $\mathbf{x}$. Every point $\mathbf{x}''$ on each surface S_i in the environment emits or reflects radiance. This radiance is then reflected at $\mathbf{x}'$ towards $\mathbf{x}$. The type of this reflection depends on the nature of the reflecting surface, which is expressed in the bidirectional reflectance distribution function (BRDF) $f_r(\mathbf{x}'' \to \mathbf{x}' \to \mathbf{x})$ [1]. Finally a geometric factor $G(\mathbf{x}'', \mathbf{x}')$ must be included that is 0 when $\mathbf{x}''$ is invisible at $\mathbf{x}'$, and is equal to $dL(dS, \mathbf{x})$ from Eq. (2-1) otherwise.

This light transfer can be expressed in the following integral equation, which is similar to the formulation of the rendering equation by Kajiya [37] ($\mathbf{x}$ can be any point in the environment):

$$L(\mathbf{x}, \mathbf{x}') = \varepsilon(\mathbf{x}, \mathbf{x}') + \int_\Omega f_r(\mathbf{x}'' \to \mathbf{x}' \to \mathbf{x}) L(\mathbf{x}'', \mathbf{x}') G(\mathbf{x}'', \mathbf{x}') d\mathbf{x} \tag{2-3}$$

1. Arrows are used here as a cue to the meaning of a BRDF. Technically f_r: $(R^3)^3 \to [0, 1]$ denotes the fraction of light incoming from $\mathbf{x}''$, reflected or transmitted on surface S that subsequently reaches $\mathbf{x}$. This relationship is determined by the reflective or transmissive properties of the surface in question.

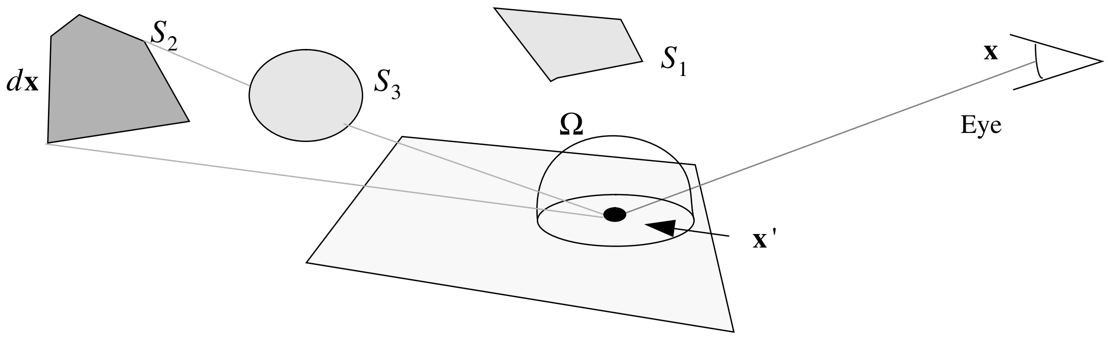


Figure 2-2. Global Illumination Calculation Geometry

Assuming that the BRDF is unaffected by heat, if there are N surfaces S_i in the environment, Eq. (2-3) can be rewritten:

$$L(\mathbf{x}, \mathbf{x}') = \varepsilon(\mathbf{x}, \mathbf{x}') + \sum_{i=1}^{N} \int_{S_i} f_r(\mathbf{x}'' \rightarrow \mathbf{x}' \rightarrow \mathbf{x}) L(\mathbf{x}'', \mathbf{x}') G(\mathbf{x}'', \mathbf{x}') d\mathbf{x} \tag{2-4}$$

Most image synthesis algorithms attempt to solve this equation using approximation techniques. In this review we concern ourselves mainly with those that attempt some form of light transport simulation, and are typically called *global illumination algorithms*. The emphasis of the first part of this thesis is a complementary one, that of understanding the nature, or *structure* of radiance functions as they vary over a surface, and developing sampling and reconstruction algorithms based on this structure. In Chapter 6, we show how these algorithms generalise to the global illumination problem.

Global illumination research has resulted in numerous algorithms that sample and reconstruct the radiance over surfaces, either for display, or in the solution of the transport problem. Thus the review of this research is important, since it relates to both sampling and reconstruction as well as the calculation of global illumination.

Solution techniques are reviewed separately in Section 2.2 that follows, while resultant sample placement and reconstruction algorithms are considered in Section 2.3. Some useful results from illumination engineering and heat transfer are then discussed in Section 2.4, and the geometry of shadows is presented in Section 2.5. Finally a discussion of previous work in the context of this thesis is presented in Section 2.6.

2.2 Solution Techniques for Light Transport

The solution algorithms developed to date can be broadly separated into four main categories. In the first approach, a system of equations is formulated and solved via a matrix solver; in the second a progressive refinement solution is used; in the third a hierarchical solution of the matrix is performed; finally the fourth is that of Monte Carlo stochastic solutions.

Heckbert and Winget [35] classify the first three as special cases of finite-element methods. The Monte Carlo or stochastic approach is significantly different, since it is an image-based sampling algorithm. The Monte Carlo light-transport solution was pioneered by Kajiya [37]. Since then a number of researchers have developed solution techniques to produce images of excellent quality (e.g., Ward, Rubinstein and Clear [69], Ward and Rubinstein [70], Shirley [53], Rushmeier and Torrance [48]). The focus of this thesis is closely related to the finite element methodology, while almost entirely distinct from the Monte Carlo solution, which will therefore not be reviewed in subsequent sections. Specifically, Monte Carlo solutions to date are mainly image-driven and view dependent; our approach constructs interpolants associated with surfaces in object space, and is used in a view-independent fashion. This distinction results in entirely different solution techniques, error metrics and overall approach to illumination calculations.

In general, the reflectance function f_r of Eq. (2-4) can include surfaces with various properties such as specularity (mirror-like reflection), anisotropy etc. For the rest of this thesis, the surfaces in the environment are assumed to be *diffuse* reflectors, so that light impinging on a surface is reflected equally in every direction. In addition the light sources in the environment are assumed to be *diffuse* emitters (i.e., emitting equally in every direction), of constant luminosity M. Light sources are assumed to possess area, although they can degenerate to a point. Monte Carlo techniques discussed above do not in general suffer from these restrictions. A number of hybrid or two-pass methods also exist, which attempt to combine the merits of the finite element approaches and the stochastic techniques, thus being able to render more general environments (e.g. Wallace, Cohen and Greenberg [68], Sillion and Puech [55], Chen et al. [11]). Although the diffuse assumption is somewhat restrictive, the problems that need to be solved are sufficiently difficult to justify this choice. In addition, this restriction will allow more rigorous studies of illumination to be performed as a first step, leading to better understanding of radiance properties.

2.2.1 Matrix Solver Solutions

In the first attempts to approximate the solution of Eq. (2-3) by Goral et al. [29] and Nishita and Nakamae [42], the surfaces in the environment were subdivided into regular grids of *patches*, over which radiosity B (a quantity related to radiance, which is simply power per area) and the reflectance function f_r were considered constant. For an environment consisting of n patches, the constant quantities radiosity, emission, reflectance and area are respectively written B_i, E_i, ρ_i, A_i, for each patch i. The geometric function $G(dA_i, dA_j)$ is rewritten as the product of $vis(dA_i, dA_j)$, a binary function expressing the visibility between the differential areas dA_i and dA_j, and the factor $dL(dA_i, dA_j)/\pi$.

By integrating over A_i on both sides of Eq. (2-4) the following expression results:

$$B_i A_i = E_i A_i + \rho_i \sum_{i=1}^{n} B_j \int_{A_i}\int_{A_j} vis(dA_i, dA_j) \frac{\cos\theta_i \cos\theta_j}{\pi R^2} dA_j dA_i, \tag{2-5}$$

where θ_i and θ_j are respectively equivalent to θ_{out} and θ_{in} in Eq. (2-1). It is important to note that B_i is a single constant scalar value, representing the assumed constant radiosity over the patch i. The double integral of Eq. (2-5) divided by an area is called a *configuration* or *form factor* in the heat transfer literature (Siegel and Howell [54]). Specifically a form factor is defined as follows:

$$F_{i \to j} = \frac{1}{A_i} \int_{A_i}\int_{A_j} vis(dA_i, dA_j) \frac{\cos\theta_i \cos\theta_j}{\pi R^2} dA_j dA_i, \tag{2-6}$$

and it expresses the fraction of the total power leaving patch i that reaches patch j. It follows from the above definition that a reciprocity relationship holds for form-factors [54]:

$$F_{i \to j} = \frac{A_j}{A_i} F_{j \to i}. \tag{2-7}$$

From equations (2-6) and (2-7), and dividing eq. (2-5) by A_i, the following system can be written:

$$B_i = E_i + \rho_i \sum_{j=1}^{n} B_j F_{i \to j}, \quad i = 1, \ldots, n. \tag{2-8}$$

Note that the direction $i \to j$ of the form-factor $F_{i \to j}$ is the opposite direction of the intuitive $j \to i$, due to the division by A_i and the application of the reciprocity equation. Eq. (2-8) can be rewritten in matrix form.

$$\begin{bmatrix} 1 - \rho_1 F_{1 \to 1} & \ldots & -\rho_1 F_{1 \to n} \\ & & \\ -\rho_n F_{n \to 1} & \ldots & 1 - \rho_n F_{n \to n} \end{bmatrix} \begin{bmatrix} B_1 \\ \\ B_n \end{bmatrix} = \begin{bmatrix} E_1 \\ \\ E_n \end{bmatrix} \tag{2-9}$$

In the original algorithm [29] the matrix of form factors was found by contour integration, for unoccluded scenes. In later work (Cohen and Greenberg [14], Cohen et al. [15]) more complex environments were considered by discretely approximating the occlusion between two patches, and thus computing an estimate for $F_{i \to j}$ in such environments. To achieve this, the form factor $F_{i \to j}$ is approximated by the differential form factor $F_{di \to j}$ which is given as follows:

$$F_{i \to j} \approx F_{di \to j} = \int_{A_j} vis\,(dA_i, dA_j)\,\frac{\cos\theta_i \cos\theta_j}{\pi R^2}\,dA_j. \tag{2-10}$$

As noted in [14], this approximation is valid when the distance R is large relative to the areas A_i and A_j of the two patches that interact, since the integrand of the outer integral in Eq. (2-5) is almost constant across A_i, and thus the outer integration does not affect the result.

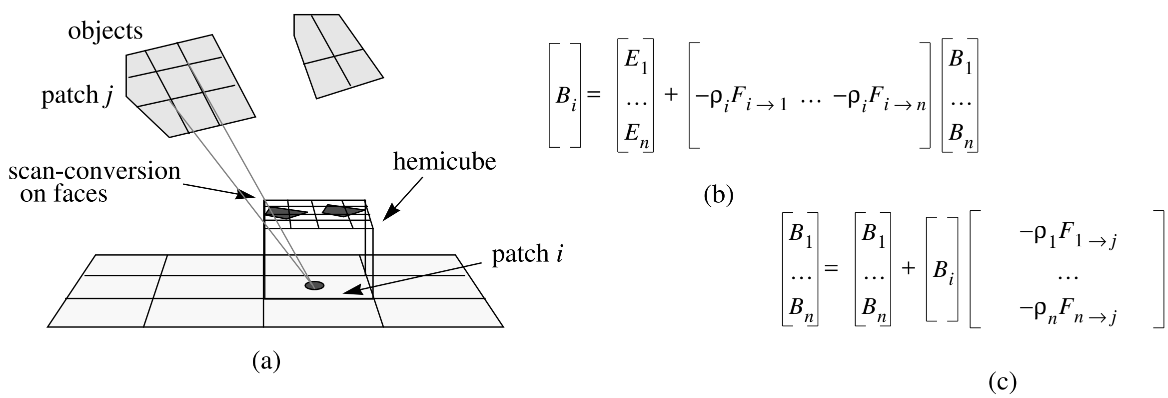


Figure 2-3. Hemicube Calculation for Form-Factors (after Cohen et al. [16])

To account for visibility, a hemicube is placed at the centre of each patch, and the rest of the environment is scan converted onto its five faces (see Figure 2-3(a)). The number of visible pixels corresponding to each patch j in the environment is taken as an approximation to the form factor $F_{di \to j}$ and thus as an approximation to $F_{i \to j}$. From Eqs. (2-8), it can be seen that this operation calculates a row of the matrix of (2-9), and represents the "gathering" of power contributed from all patches j to the collecting patch i (Figure 2-3(b)).

Once the matrix was formed, the system Eq. (2-9) was solved using an iterative matrix solver. This method is generally called the *full matrix* radiosity-based approach.

2.2.2 Progressive Refinement

In an environment containing n patches, the full matrix approach requires the calculation and storage of $O(n^2)$ form factors. To alleviate this problem, a *progressive refinement* approach was adopted by Chen [10] and Cohen et al. [16]. Intuitively, the progressive refinement approach calculates the contribution of one "shooting" patch i to all the other patches in the environment, reversing the process of "gathering". Using the reciprocity relationship for form-factors, the contribution of shooting patch i to receiving patch j, can be written as

$$B_{j \leftarrow i} = \rho_j B_i F_{j \to i} = \rho_j B_i F_{i \to j} \frac{A_i}{A_j}, \qquad j = 1, \ldots, n. \tag{2-11}$$

Given this formulation, the same hemicube algorithm can be used to account for the visibility factor, and thus to compute the form factors $F_{j \to i}$. From (2-11), it can be seen that a column of the matrix of Eq. (2-9) is being computed for each "shot" from each patch i (Figure 2-3(c)). In addition, it can be seen that after every iteration expressed by (2-11), every other patch has received some contributed illumination, and thus an initial image can be generated. Conversely, the full-matrix solution requires the computation of the whole matrix to generate any image.

Gortler, Cohen and Slussalek [28], show that the progressive refinement algorithm is essentially an iterative relaxation method. This formulation is discussion in more detail in Sections 6.2.3 and 6.3.2.

Both the progressive refinement and the full matrix approaches require a mesh of patches that is a regular discretisation of a polygonal environment. The mesh is then rendered by extrapolation of the constant radiosity values on each patch, to its vertices, allowing the rendering of smoothly shaded polygons. Bilinear interpolation or Gouraud shading [24] are generally used. This final stage will be referred to as the *display stage* in global illumination algorithms.

2.2.3 Hierarchical Solution Method

As mentioned above, it is necessary to compute the complete matrix of form-factors to solve the system Eq. (2-9), via a matrix solver algorithm. Hanrahan and Salzman in [31] and Hanrahan, Salzman and Aupperle [32], observed that many elements of the matrix are very small, and thus using the hemicube calculations for each form factor often results in much higher accuracy computations than necessary, and consequently in wasted computation time.

To take advantage of this fact, a hierarchical approach is used. A quadtree of patches is built on each polygon, and patches of one polygon are linked to patches of another if there is light transfer between them. Adaptive refinement is then performed if the form factor between two linked patches is larger than a preset tolerance. If the form factor is less than the tolerance, the two subpatches are linked and the refinement ends. At the end of the refinement process, the power is transported across the links. The trees of each polygon are traversed in depth-first fashion, first pushing the radiosity values from the parent nodes to the children, and then pulling them back up the tree, so that each node receives a radiosity value equal to the area-weighted average of its children.

The original approach [32] added two modifications to this basic algorithm, first by performing the refinement iteratively, effecting a multigridding approach, and a second in which the refinement criterion is based on potential power transfer, and not only on form-factor accuracy.

One problem with the approach is the necessity to build all initial links between the top-level patches, that is the input polygon set. To overcome this problem Teller and Hanrahan [61] preprocess the environment using the concept of “conservative triage”: efficient algorithms are used to classify patch pairs as visible, invisible or partially visible. In the last case, processing is delayed until subdivision renders the effect of partial visibility unimportant. In addition, the algorithms presented allow the maintenance of potential blocker lists between patch pairs.

2.3 Resampling for Illumination Reconstruction

The basic solution methods described in Section 2.2 have been refined in subsequent research to achieve higher quality or faster solutions. A number of these advances have led to different methods of sampling and reconstructing radiosity or radiance functions defined over receiver surfaces. Some of these were motivated by the need to improve visual image quality, and others to achieve faster or more accurate solutions to the system of equations for light transport.

2.3.1 Adaptive Subdivision

In the original radiosity-based solutions, patches tended to be too large to capture fine image detail, especially close to shadow boundaries, and also in regions in which the radiosity had a high gradient. As a remedy to this problem, Cohen et al. [15] developed an adaptive subdivision algorithm. In this approach, patches are subdivided into *elements* that are used for display, while the larger patches are used for the actual solution process. Thus, element-to-patch form factors are computed using the hemicube when necessary, but only patch-to-patch form factors are included in the matrix and the actual matrix solution.

Elements are also used by Cohen et al. [16] for progressive refinement. There are problems with this approach however, since the approximation of Eq. (2-10) is based on the assumption that patch i is small compared to patch j and the distance between them. This no longer holds true when using the hemicube for progressive refinement, since a large patch i shoots light power to small elements. This is particularly true when surfaces are close together. This was noted by Baum, Rushmeier and Winget [6], and analytic form factor calculations were performed when deemed necessary, using Eq. (2-26) (see below).

Wallace, Elmquist and Haines [67] introduced ray-tracing as an alternative to the hemicube visibility computation. This allowed the direct computation of the light transfer to the element vertices of a grid, by point sampling the source, and computing an approximation to the differential form factor:

$$dF_{i \to dj} = \frac{dA_j}{A_i} F_{di \to j}. \qquad \text{(2-12)}$$

Many hemicube aliasing artifacts were eliminated in this way. Adaptive subdivision of the source was necessary in some shots, to achieve higher accuracy computation of the form-factors $dF_{i \to dj}$.

For the hierarchical solution process, Smits, Arvo and Salesin [57] add *importance* as an additional criterion to form-factor accuracy in the refinement step of the hierarchical algorithm. Importance can be determined by the adjoint to the radiosity equation and thus propagated in the same step as radiosity. Intuitively, importance-driven subdivision enforces a finer mesh in regions whose power emission is more likely to reach the eye, and a much coarser mesh in areas where the light power transmission will probably not reach the viewer.

In all the above methods, adaptive element or patch subdivision results in a regular, non-uniform, piecewise constant interpolant grid over the surfaces in the environment.

Campbell and Fussell [9] present the only previous work in which structure of illumination is taken into account. From observation they assume that radiance functions in unoccluded regions display only one maximum, while in the penumbra multiple maxima and minima exist. Numerical optimisation is used to determine these extrema, and subdivision is performed at the midpoints between extrema. The resulting grid is a piecewise constant triangular element grid, based on discontinuity meshes (see Section 2.5).

2.3.2 Higher Order Reconstruction and Solution Methods

The fact that the piecewise constant interpolants constituted an important limitation to the radiosity-based techniques has become apparent in recent research. In particular, Tampieri and Lischinski noted this limitation in [59] calling it the “constant radiosity assumption syndrome”. Subsequent research has removed the restriction of piecewise constant radiance representation.

In general, an approximation $\tilde{B}(\mathbf{x})$ to the radiosity or radiance function $B(\mathbf{x})$ over patch i can be represented by a weighted sum of K basis functions:

$$\tilde{B}_i(\mathbf{x}) = \sum_{k=1}^{K} B_i^k h_i^k(\mathbf{x}) . \qquad (2\text{-}13)$$

This formulation allows radiosity to be represented by arbitrary order basis functions, including higher degree polynomials. The advantages of such higher order representations over piecewise constant approaches include smoother representation of the radiosity function, consistent representation across solution and display stages, as well as potentially faster convergence in certain solution techniques.

Lischinski, Tampieri and Greenberg used quadratic triangular interpolants in [39] (described in more detail by Tampieri [60]). Salesin, Lischinski and DeRose [49] used cubic reconstruction over triangular elements to represent radiance. To this end a Clough-Tocher cubic polynomial was built over triangles, and shadow discontinuities were subsequently represented by relaxing constraints over shadow boundaries. Non-polynomial approximate reconstruction has been used by Vedel [66], where a truncated Gaussian filter is employed to reconstruct intensity.

Heckbert and Winget [35] cast Eq. (2-3) in a standard integral equation form:

$$B(\mathbf{x}) = E(\mathbf{x}) + \int_{\Omega} \kappa(\mathbf{x}, \mathbf{x}') B(\mathbf{x}') d\mathbf{x}', \tag{2-14}$$

where $\kappa(\mathbf{x}, \mathbf{x}')$ is called the *kernel* of the integral equation, and is expressed as follows:

$$\kappa(\mathbf{x}, \mathbf{x}') = f_r(\mathbf{x}, \theta_1, \theta_2)\, vis(\mathbf{x}, \mathbf{x}') \frac{\cos\theta_1 \cos\theta_2}{\|\mathbf{x} - \mathbf{x}'\|^2}, \tag{2-15}$$

where the angles θ_1, θ_2 are equivalent to the angles $\theta_{in}, \theta_{out}$ in Eq. (2-1). The kernel is thus essentially the general case of the differential form-factor with occlusion, multiplied by the reflectance function.

Standard *projection* methods can then be used to formulate a system of equations when radiosity is represented by basis functions, resulting in higher order solution algorithms. Heckbert and Winget continue by proposing the use of the method of weighted residuals and specifically Galerkin methods for “flatland” (two-dimensional space). Tests of these techniques for two dimensions were presented by Heckbert in [34], where linear and quadratic interpolants appeared to give faster convergence. Heckbert notes the importance of discontinuity meshes discussed below.

This introduction of the problem has led to three separate three dimensional applications of Galerkin methods by Zatz [72], Troutman and Max [64] and Gortler et al. [27]. A brief overview of the mathematics necessary to formulate the system of equations, using an orthonormal basis function set, is presented next. Notation is taken mainly from [72] and [27]. A more general presentation of the Galerkin method can be found in Section 6.2.2.

By parameterising the surfaces in the environment in s, t or u, v, all in $[-1, 1]$ and solving for the approximation to radiosity $\tilde{B}(\mathbf{x})$ instead of the actual function $B(\mathbf{x})$ in (2-14), the following expression can be derived:

$$\tilde{B}_i(s,t) = \tilde{E}_i(s,t) + \sum_{j=1}^{n} \int_{-1}^{1} \int_{-1}^{1} \kappa_{ij}(s,t,u,v)\, \tilde{B}_j(u,v)\, du dv, \tag{2-16}$$

where n is the total number of patches in the environment. Eq. (2-16) can be rewritten:

$$\sum_{k=1}^{K} B_i^k h_i^k(s,t) = \sum_{k=1}^{K} E_i^k h_i^k(s,t) + \sum_{j=1}^{n} \int_{-1}^{1} \int_{-1}^{1} \kappa_{ij}(s,t,u,v) \sum_{m=1}^{M} B_j^m h_j^m(u,v)\, du dv. \tag{2-17}$$

Now, define an inner product operator on two functions, where $W(s,t)$ is a weighting function:

$$\langle f, g \rangle_W = \iint f(s,t)\, g(s,t)\, W(s,t)\, ds dt. \tag{2-18}$$

By taking the inner product with all the basis functions, assuming that the functions h are orthonormal, i.e. $\langle h^k, h^l \rangle = 0, k \neq l$ and $\langle h^k, h^l \rangle = 1, k = l$, and using Eq. (2-17), the following expression can be written for the coefficient of the kth basis function of the interpolant built on patch i:

$$B_i^k = E_i^k + \sum_{j=1}^{n} \sum_{l=1}^{L} B_j^l \left\langle \int_{-1}^{1} \int_{-1}^{1} \kappa_{ij}(s,t,u,v)\, h_i^l\ du dv,\ \ h_j^k \right\rangle_W. \tag{2-19}$$

By writing the inner product on the right hand of Eq. (2-19), as K_{ij}^{kl} (which is the equivalent of a form-factor in eq. (2-8)),

$$K_{ij}^{kl} = \left\langle \int_{-1}^{1} \int_{-1}^{1} \kappa_{ij}(s,t,u,v)\, h_i^l\ du dv,\ \ h_j^k \right\rangle_W, \tag{2-20}$$

the following system can be written:

$$B_i^k - E_i^k = \sum_{j=1}^{n} \sum_{l=1}^{L} B_j^l K_{ij}^{kl}. \tag{2-21}$$

The formulation of Eq. (2-21) is thus in a standard matrix form and can be solved by traditional matrix solver techniques, once the K_{ij}^{kl}'s have been computed.

Legendre and Jacobi polynomials are used by Zatz [72], while Gortler et al. [27] use a hierarchical wavelet basis. For visibility, Zatz intersects the whole environment is intersected for the calculation of each K_{ij}^{kl}, while Gortler et al. use jittered ray-casting. Troutman and Max [64] use a hemicube for linear elements, and an a priori, interactively specified, discontinuity mesh for higher order elements.

In the approaches described by Zatz [72] and also by Gortler et al. [27], it is not clear if the specific basis function chosen are appropriate for radiance (or radiosity) reconstruction. Wavelets are not smooth across boundaries, introducing visual artifacts, while Jacobi/Legendre polynomials can produce significant ringing artifacts. As noted by Gortler et al. in [27], continuous splines may be more appropriate.

2.4 Results from Illumination Engineering and Heat Transfer

Illumination engineering researchers and heat transfer engineers have developed a large number of useful techniques and mathematical derivations concerning light calculations. In this thesis several of these will be used, and it is therefore convenient to present them here.

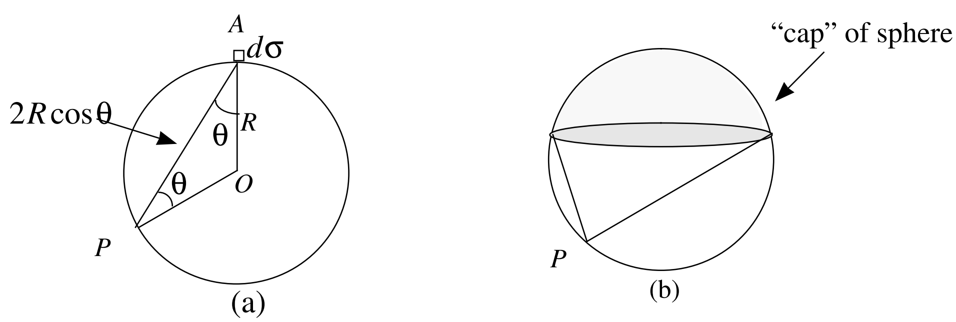


Figure 2-4. Isolux Sphere Geometry (after Parry [40])

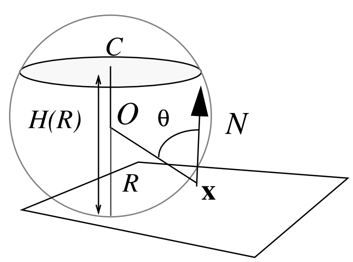
Consider a sphere, on which a differential emitting element $d\sigma$ is placed. A cross-section of this sphere is shown in Figure 2-4(a). From Eq. (2-1), it can be seen that the illumination at the point P is given as follows (Parry [40]):

$$dL(d\sigma, P) = M\frac{\cos\theta\cos\theta}{(2R\cos\theta)^2}d\sigma = M\frac{d\sigma}{R^2}. \tag{2-22}$$

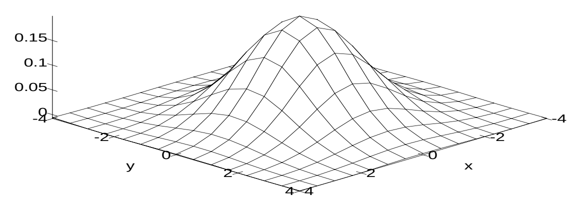
Thus the radiance at any point P on the sphere is independent of the position of P on that sphere. Such a sphere is called an *isolux sphere*. In addition, Parry shows that if a number of these elements are “fused” together to form a “cap” on the sphere shown in Figure 2-4(b), the illumination at any point P on the sphere remains constant. He also shows that the illumination from a disc source is equal to that of a cap formed by the portion of the sphere it cuts away (see the shaded area above the disc in Figure 2-4(b)).

Given the above, consider the isolux sphere S with radius R defined by the disc and the point $\mathbf{x}$, and has its centre at O. $H(R)$ is the distance from the disc to the point on the sphere directly opposing the centre (see Figure 2-24(a)) and is proportional to R. If the disc has radius a, and luminosity M, and θ is the angle between the normal to the receiver N and the line $O\mathbf{x}$, as shown in Figure 2-24(a) then the radiance at any point on the sphere is [40]:

$$L_{S(disc)}(\mathbf{x}) = M\cos\theta\frac{a^2}{H(R)^2 + a^2}. \tag{2-23}$$



(a)



(b)

Figure 2-5. Disc Emitter Geometry

For any source S, with luminosity M, Eq. (2-2) can be rewritten as follows:

$$L_S(\mathbf{x}) = M\int_S \frac{\cos\theta_{out}\cos\theta_{in}}{R^2}dS = M\int_S \frac{\vec{R}\cdot\vec{N}}{R^2}(\vec{R}\cdot\vec{N}_S)\,dS = M\int_S \left(\frac{\vec{R}\cdot\vec{N}}{R^2}\vec{R}\right)\vec{N}_S\,dS, \tag{2-24}$$

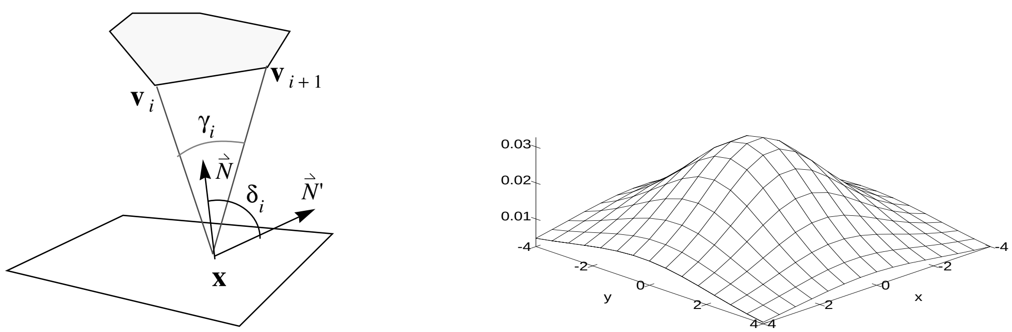
where $\vec{N}_S$ is the normal vector of the emitter polygon. The right hand side of Eq. (2-24) is in a form for which Stokes theorem can be applied giving a contour integral expression (after Vedel [65]):

$$L_S(\mathbf{x}) = M \oint_{C_S} \frac{\vec{N} \times \vec{R}}{R^2} d\vec{C}_s, \tag{2-25}$$

where C_S is the contour of the source and $d\vec{C}_S$ is a differential element of the contour. In the special case of a polygonal emitter, $d\vec{C}_S$ is a vector that is piecewise constant along each edge of the polygon. Thus for a polygon emitter with constant luminosity M, the radiance at a point $\mathbf{x}$, referring to Figure 2-6(a) is:

$$L_{S(polygon)}(\mathbf{x}) = M \sum_{i=1}^{n} \gamma_i \cos \delta_i, \tag{2-26}$$

where the quantity γ_i is the interior angle formed by the point $\mathbf{x}$ and the emitter vertices $\mathbf{v}_i$, $\mathbf{v}_{i+1}$, and δ_i is the angle of the normal $\vec{N}'$ to the plane defined by $\{\mathbf{x}, \mathbf{v}_i, \mathbf{v}_{i+1}\}$ and the receiver normal $\vec{N}$.



(a) (b)

Figure 2-6. Polygon Emitter Geometry

For any emitter, the radiance function over a receiver can be plotted as a height field. Such plots are shown in Figure 2-24(b) for the disc emitter and Figure 2-6(b) for the polygonal emitter, over a receiver at a slight angle.

2.5 Umbra, Penumbra and Discontinuity Meshes

Shadows give important cues on depth, relative positioning and lighting conditions. From the very early days of computer graphics, generating some form of shadows has been an important goal (e.g., Bouknight and Kelley [7], Appel [3], Atherton, Weiler and Greenberg [4]).

In addition, in the context of global illumination, shadow boundaries represent discontinuities of various kinds in the radiance function across a surface. Their identification is thus important in improving the accuracy and the speed of the light transport solutions, as well as for sampling and reconstruction for display. In the approaches reviewed in Section 2.2 and 2.3, these discontinuities are sampled implicitly for the most part during the solution process, as a result of adaptive discretisation. In general, occlusion computations represent the largest portion of all calculation time for most global illumination algorithms. This is due to the required intersection of the entire scene to determine occlusion, using either the hemicube algorithm or ray-tracing.

2.5.1 First Shadow Research in Computer Graphics

The early days of image synthesis saw the use of point or directional light sources, that rarely occur in real scenes. Both cast sharp shadows, and thus only create areas of darkness (umbra) and light. Modelling more common light sources such as linear and area sources was attempted in a number of cases in ray-tracing algorithms (e.g. Amanatides [1], Poulin and Amanatides [46], Tanaka and Takahashi [58]). These algorithms allowed the rendering of regions of partial illumination (penumbra).

2.5.2 Aspect Graphs and Geometric Visibility

An important body of work exists in the computer vision literature that is closely related to the computation of shadow boundaries. Gigus and Malik [25] present an algorithm in which the viewing space is partitioned into a set of equivalent views. This problem is closely related to the determination of areas in which the visible region of a source are equivalent, which is exactly the problem of finding penumbral boundaries. In this paper and in subsequent work by Gigus, Canny and Seidel [26], visual events were categorised and investigated in detail.

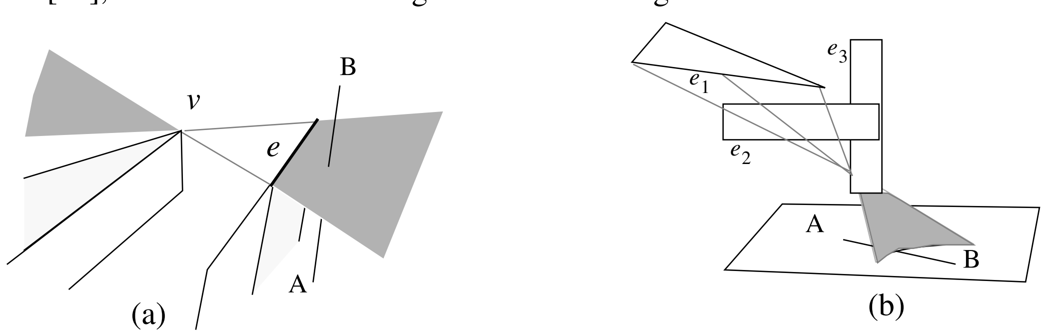


Figure 2-7. Visual Events and Corresponding Discontinuity Surfaces

Changes in visibility are caused by the interaction between edges and vertices of the objects in the environment, called visual events. Visual events are of two main types:

- EV events: which are caused by the interaction of a vertex and an edge; and
- EEE events: which are caused by the interaction of three edges in environment.

An EV event is shown in Figure 2-7(a) and a EEE event is shown in Figure 2-7(b). In both, the visual event occurs when crossing from A to B. For the EV event, visibility changes only when crossing the plane formed by edge e and vertex v in the shaded regions of Figure 2-7(a). When this plane is crossed from A to B, vertex v becomes visible, while when crossing from B to A, vertex v becomes invisible.

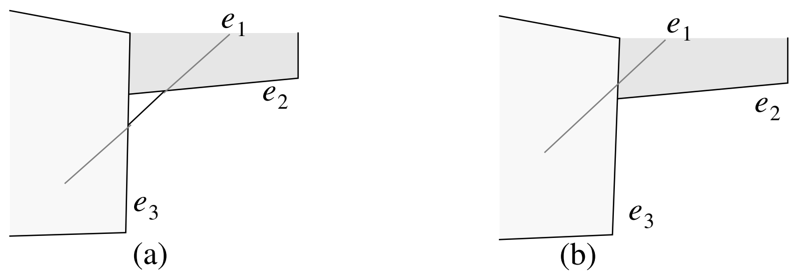


Figure 2-8. Visibility Changes due to a EEE Surface

For the EEE case, three edges $\{e_1, e_2, e_3\}$ form a quadric surface. When crossing this quadric surface, the visibility changes. The visibility change that occurs when crossing a EEE surface is the appearance or disappearance of an edge. When looking from A (in Figure 2-7(b)) to e_1, the edge e_1 is visible (see Figure 2-8(a)), while when looking from B (in Figure 2-7(b)) to e_1, it is not (Figure 2-8(b)). The plane defined by $\{e, v\}$ and the quadric surface defined by $\{e_1, e_2, e_3\}$ are called *discontinuity surfaces*.

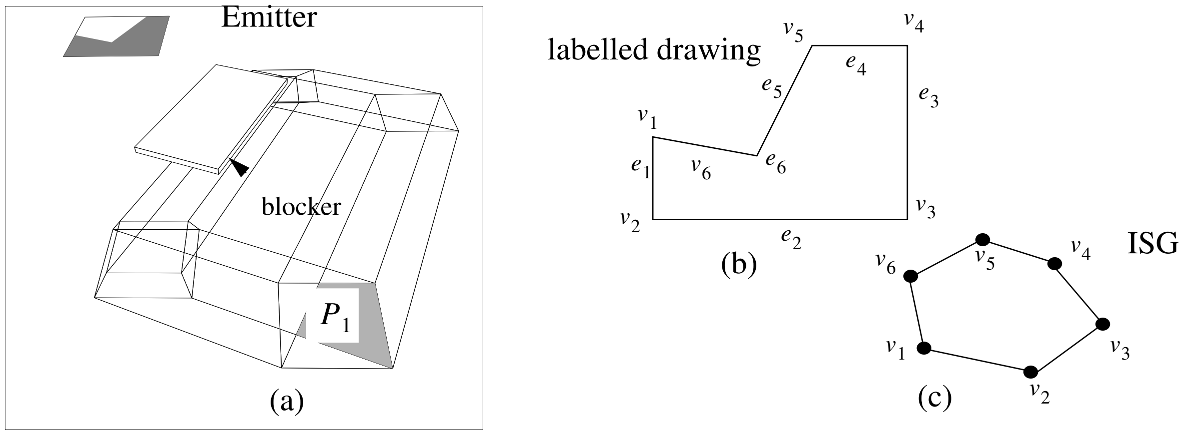


Figure 2-9. The Aspect of a Point

Consider the environment shown in Figure 2-9. The visible region of the emitter from point P_1 is shown shaded on the emitter. If the edges and vertices of the polygons of this region are labelled, a labelled line drawing is created as defined in [25] (Figure 2-9(b)). A labelled graph, called an image structure graph (ISG), can be constructed by creating a vertex in the graph for each vertex of the line drawing and an edge for each line segment in the labelled line drawing, again following the construction of [25] (Figure 2-9(c)). The *aspect* of a point is the topological graph structure of the ISG of that point. If the ISG's of two viewpoints are isomorphic, then their aspects are the same. In the terminology of Stewart and Ghali [56], the aspect of a viewpoint with respect to an emitter σ is the σ-*aspect* of the viewpoint.

2.5.3 Discontinuity Meshing in Computer Graphics

One way to deal with shadow boundaries and resulting discontinuities, is to compute them explicitly. For the purpose of this review the *complete discontinuity mesh* for a specific emitter σ is considered a mesh such that in each face of the mesh, the σ-aspect of every point is the same. To generate such a mesh all EV and EEE surfaces that interact with the emitter have to be computed. The *extremal* boundary of the penumbra is the boundary separating completely unoccluded regions from regions of partial occlusion. The extremal umbral boundary is the boundary between penumbra and umbra. More detailed definitions are given in Chapter 4.

Extremal boundaries were computed early on by Nishita and Nakamae [41]. In this approach, extremal umbral and penumbral boundaries were computed for simple geometries and exact illumination was computed by explicitly calculating the visible region of the source from every point.

Campbell and Fussell [8] approximated area sources by collections of point light sources. Binary Space Partition (BSP) trees were used to achieve a front-to-back traversal of the environment at the price of significant splitting of polygons in the environment. This approach was extended by Campbell and Fussell [9] to consider area light sources explicitly. The extremal penumbral boundary was completely determined by union of the BSP tree volumes corresponding to each blocker encountered. Extremal umbral boundaries were not always correctly found however, since they would require the calculation of quadric EEE surfaces. The resulting mesh was represented as a two-dimensional BSP tree on each receiving surface, and then adaptive subdivision was performed using the numerical optimisation techniques discussed briefly in Section 2.3.1. Very similar mesh computations were performed by Chin and Feiner [13] (extended from their earlier paper [12], in which point sources were treated).

EV surfaces other than extremal were first calculated by Lischinski, Tampieri and Greenberg [39], using BSP trees for front-to-back traversal of the scene. Only EV surfaces containing a source edge or vertex were considered, and thus the faces or elements of the mesh usually contained regions with different σ-aspects. The approach was combined with the hierarchical solution of [32] in recent work by the same authors [39]. The hierarchical quadtree on each polygon is modified, and instead of a regular structure, subdivision occurs along certain discontinuity lines of the primary light sources. After the global hierarchical computation, the modified quadtree mesh is triangulated using a constrained Delaunay triangulation. Finally a “local pass” is added, in which visibility can be recomputed in the triangular mesh elements, depending on the quality desired.

Heckbert in [34] performed a detailed study of discontinuity meshing in two dimensions. In this study, every pair of edges and vertices is visited to generate a complete flatland discontinuity mesh, with cost N^3, where N is the number of edges in the environment. The work was extended to three dimensions in [33]. In a fashion similar to the flatland algorithm, Heckbert proposes that every EV surface be traced, although he ignores EEE surfaces. In the implementation presented, only EV surfaces with an edge or vertex of the source were actually traced, and only for simple scenes.

Teller in [62] performs a related computation, in which visible regions of a source through a sequence of portals were computed. The computation is exactly equivalent to correctly computing the extremal umbra boundary for such configurations. This is the first treatment of EEE surfaces. The approach however uses an involved five-dimensional representation in Pluecker coordinates and according it is not straightforward to robustly generalise to environments other than the portal configurations described (Teller and Hanrahan [61]). In addition the focus of the paper is in the efficient determination of extremal boundaries, rather than the complete mesh.

The first algorithm that can compute the complete mesh was proposed by Stewart and Ghali in [56]. In this approach, a plane parallel to the emitter is swept across the environment. At every peak vertex in the environment (i.e., a vertex in which all incoming edges point "down" with respect to the source), the σ-aspect is built for that vertex. At every other vertex in the environment, and at the intersection of all relevant EV surfaces and EEE surfaces with the edges of the environment, the aspect is updated incrementally. The algorithm provides the benefit of robustness and guaranteed good asymptotic complexity behaviour, at the cost of complicated data structures and algorithms for the plane sweep and the aspect update in three dimensions on the sweep plane.

2.6 Discussion

The radiosity-based approach has resulted in sophisticated techniques for the determination of radiosity or radiance across surfaces for image synthesis. Most of the advances have been motivated by the light transfer process rather than achieving high quality reconstruction for the display stage. Partially as a result of this bias, the nature or *structure* of the functions involved is rarely studied or used, even though for the most part only diffuse environments have been considered.

In the most closely related previous work, Campbell and Fussell [9] observe that unoccluded illumination is unimodal, on a purely empirical basis. A precise treatment of this concept is presented in Chapter 3 of this thesis. This behaviour of radiance functions is defined carefully, and shown to be true for disc sources, while the class of convex polygonal sources is conjectured to have this property. In addition curvature of the radiance function is identified as another important property for sampling and reconstruction. The resulting sampling and reconstruction algorithms presented allow more consistent, accurate and efficient representation of radiance for unoccluded environments.

Campbell and Fussell [9] also observe that radiance in penumbral regions appears to have multiple maxima and minima, and resort to numerical optimisation to determine them. The cause of this behaviour was not investigated further by them. Lischinski, Tampieri and Greenberg [38], claim that finding additional discontinuities alleviates the problem of sampling these regions. In Chapter 4 of this thesis it is shown that computing the complete mesh and σ-aspects is essential in comprehending the behaviour of radiance in partially occluded regions. An algorithm to compute the complete mesh and σ-aspects is also presented, which has been partially implemented with very encouraging results, facilitating the understanding of radiance in penumbral regions and the development of algorithms to sample and reconstruct it. The only other complete mesh computation algorithm [56] has been proposed concurrently with the research in this thesis, with the trade-offs mentioned in Section 2.5.

In all previous work, the structure of radiance in regions of partial occlusion has been considered either completely unknown, or only possible to determine numerically, due to lack of the knowledge of radiance function structure. By combining the understanding of radiance acquired by the research of Chapter 3, and using the framework and algorithms of Chapter 4, empirical studies of radiance in penumbral regions has been performed in Chapter 5. It was observed that a majority of faces display monotonic and smooth radiance, and that many of the discontinuities can be ignored for the purposes of approximation. Algorithms to reduce the size of the mesh and to select the degree of the interpolants for penumbral regions are developed, based on these observations.

The research of Chapters 3 to 5 allows the efficient creation of an accurate and compact representation of radiance due to a single emitter. As noted by Heckbert in [34], higher-order (than constant) representations of radiance should be used in conjunction with discontinuity meshing to achieve improved results in the solution of the light transfer equation (2-4). The research of this thesis facilitates this by developing algorithms that produce a higher order representation (a combination of constant, linear and quadratic interpolants), with the use of discontinuity meshes.

Embedded in this representation are tight and relevant error bounds of the approximations employed. In previous work, due to the lack of consideration of the structure of radiance functions, error bounds were often impossible to determine or based on averages of dubious utility. In Chapter 6, the structured sampling and reconstruction algorithms are applied to the more general settings of multiple sources and global light transfer using an iterative Galerkin method. Desired accuracy criteria, such as those of the hierarchical solution [32], and the ability to propagate light power by first processing the most important interactions as in the progressive refinement solutions [10][16] can now be combined in a meaningful and accurate fashion. This contrasts with recent higher order solutions in which sometimes the benefits of higher order representation are lost due to neglect of important visual considerations (e.g., the choice of wavelet basis results in C^0 discontinuities in [27] that are visually unacceptable), or somewhat mechanistic application of solution methodologies (e.g., direct application of Galerkin methods without analysis of resulting visual quality resulting from the choice of basis functions in [72] and [27]).

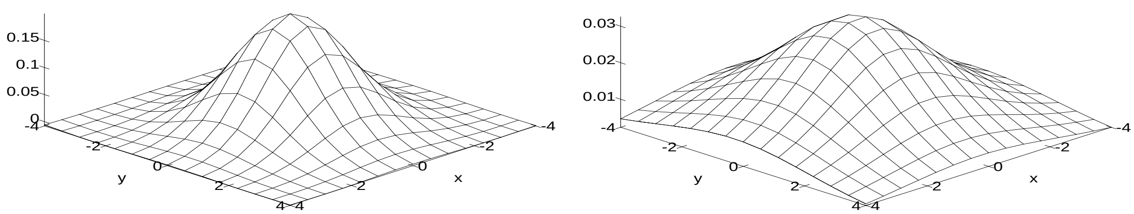
Chapter 3

Structured Sampling for Unoccluded Area Sources

To render an image illuminated by an unoccluded area light source, it is necessary to calculate the value of radiance at each visible pixel. If only direct illumination is considered, the highest quality image can be created by evaluating the analytic formula of radiance from a source, such as equations (2-23) or (2-26) for a disc and polygonal source respectively. In general this is an expensive solution, and it is much more efficient to represent illumination by a simple approximating function. Such a representation has many other benefits, especially in the global light-transfer solution presented in Chapter 6, in which it is necessary to evaluate and integrate radiance on surfaces many times until convergence.

To construct a good approximation of radiance functions, it is necessary to first examine the nature or *structure* of illumination from an unoccluded area source. Previous approaches have used regular, uniform or adaptively subdivided grids to sample the function and build interpolating functions (see Section 2.3). In this chapter a new, accurate, consistent and efficient sampling and reconstruction algorithm is presented. The approach is accurate because it guarantees tight and relevant error bounds on the piecewise continuous segments of the interpolant. It is consistent since it tracks important properties of the radiance function in animated sequences thus avoiding annoying visual artifacts. Finally it is efficient since the number of samples of the radiance function required to represent radiance at the same quality is greatly reduced compared to previous approaches.

We shall first discuss important properties of radiance. Based on these properties, a strategy is developed to construct an interpolating representation of radiance. An efficient algorithm that reduces the analysis of two dimensional radiance functions to the analysis of simpler, one-dimensional functions is presented next. The construction of a two-dimensional interpolant using this analysis is then described. Implementation and experimental results are finally presented and analysed in detail.



(a) (b)

Figure 3-1. Disc and Polygonal Source Radiance Functions

3.1 The Nature of Radiance Functions

Radiance functions from diffuse area light sources on diffusely reflecting receivers are in general smoothly varying functions, punctuated by shadow and surface discontinuities. In this chapter we shall only consider unoccluded environments. In Chapter 2, radiance functions over a simple diffuse receiver were shown for two area sources, a disc and a polygonal source. These plots are repeated here in Figure 3-1(a) and (b) respectively. These figures show that the functions seem to have one maximum and a slow drop off to zero away from that maximum. This corresponds to intuition, since both functions, defined by equations (2-23), and (2-26), contain terms that decrease with distance (the terms $1/H(R)^2$ for the disc and γ_i for the polygon).

In the following discussion it will be useful to consider the restriction of the two-dimensional radiance function on a linear subdomain. An example is shown in Figure 3-2(b), which is a plot of the radiance function due to the source shown in Figure 3-2(a), along line *l*.

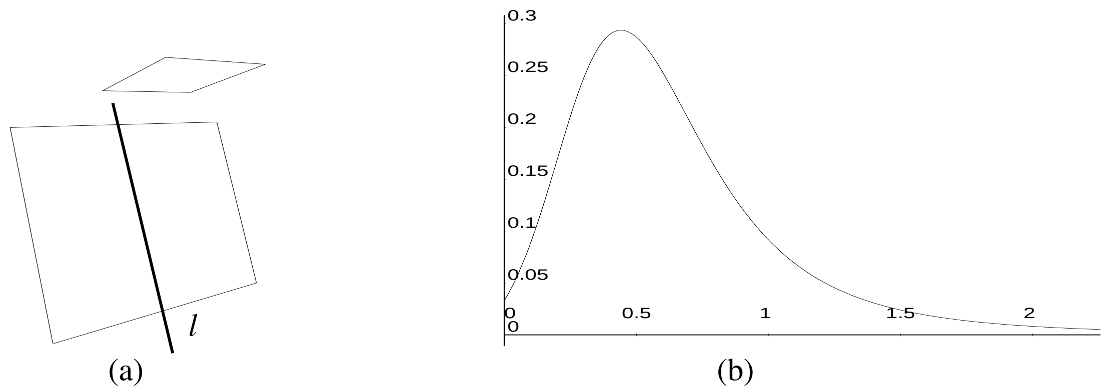


Figure 3-2. Rectangular Source and Corresponding Illumination on a Linear Subdomain

3.1.1 Unimodality

The intuitive property of smooth drop off from a single maximum can be formally described by the property of unimodality. The existence of a single maximum was empirically observed first by Campbell and Fussell in [9], but was not further investigated.

Definition 3-1: *A function* $f(x, y)$ *is unimodal if and only if the function has only one maximum, and the restriction of* $f(x, y)$ *to any linear subdomain* $ax + by + c = 0$, *also has only one maximum and is monotonically non-increasing as a function of distance from that maximum.*

This property does not hold for all polygonal sources. It can be shown that concave polygonal sources may have multiple maxima. Consider for example the configuration shown in Figure 3-3(a), where the source consists of two triangular elements connected by a thin rectangle.

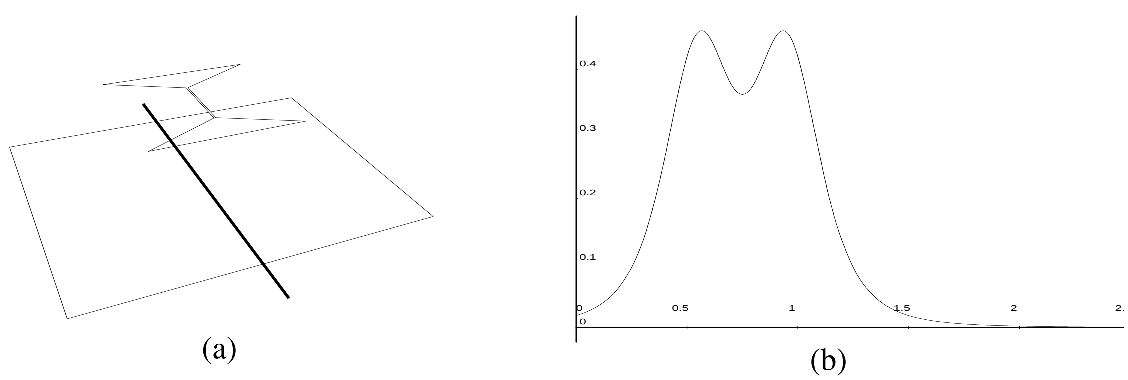


Figure 3-3. A Non-Unimodal Polygonal Source

The illumination along the line directly under the thin rectangle connecting the two triangles is shown in Figure 3-3(b). It exhibits two maxima, and therefore is not unimodal. On the other hand, the illumination of each individual triangle is unimodal.

For simpler geometries, more precise observations can be made. Specifically for a disc source illuminating an arbitrarily oriented receiver, unimodality can be proven.

3.1.1.1 Unimodality of Radiance due to a Disc Light Source

The radiance at a point (*x,y*), due to a disc light source of radius *a*, with constant luminosity *M* was given by Eq. (2-23), repeated below:

$$L_{S\,(disc)}\,(x, y) \;=\; M\cos\theta\frac{a^2}{H\,(R)^2 + a^2} \tag{3-1}$$

The sphere formed by three points on the disc and the point $P\,(x, y)$, on surface *S*, has radius *R*. The sphere and the distance *H(R)* are shown in Figure 3-4(a), while the angle θ is the angle between $\overline{OP}$ and the normal $\overline{N}$ to the surface *S*, as shown in Figure 3-4(b).

Theorem 3.1: *For any disc light source D_s centred at C, with radius a, and outgoing unit normal N_s, and for the receiver plane $z = 0$, the radiance function $L\,(x, y)$ is unimodal.*

Proof: To prove the theorem, the equation is split into the $a^2/(H\,(R)^2 + a^2)$ term and the cosine scaling factor. Unimodality is shown for the first term, and then it is shown that the cosine scale factor preserves this property.

All the isolux spheres defined by the disc D_s and any point *P* on the receiver, have centres that are positioned on the line $L\,(t) = C + tN$ [40] (see Figure 3-4(b)). Define $S\,(t')$ to be the sphere centred at $O = L\,(t')$ for a given $t = t'$ (see Figure 3-4(b)). Call $R_s\,(t')$ the radius of this sphere.

Lemma 3-1: *There exists only one value t_{min} of t on the line $L\,(t)$ such that the corresponding unique sphere $S\,(t_{min})$ is tangential to the plane $z = 0$ at point P_{max}, and thus has only one point in common with the plane.*

Proof: See Appendix A.1.

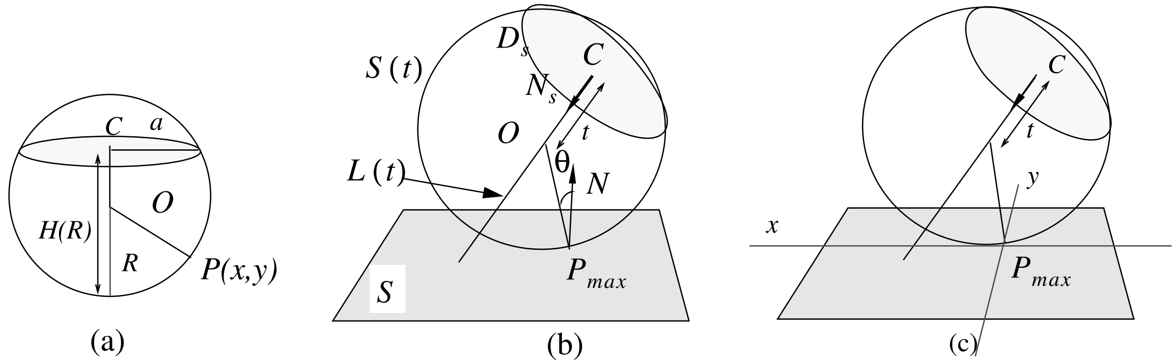


Figure 3-4. Unimodality Proof for a Disc Source

By substituting the above terms into Eq. (3-1), we find that there are two possible tangential points, and that only one is relevant since the second point is behind the source and is not illuminated. Details of the proof of Lemma 3-1 can be found in Appendix A.1.

Any value t less than t_{min} results in a sphere $S(t)$ that does not intersect the receiving plane. The radius $R_s(t)$ of any such sphere is given as follows (see also Appendix A.1):

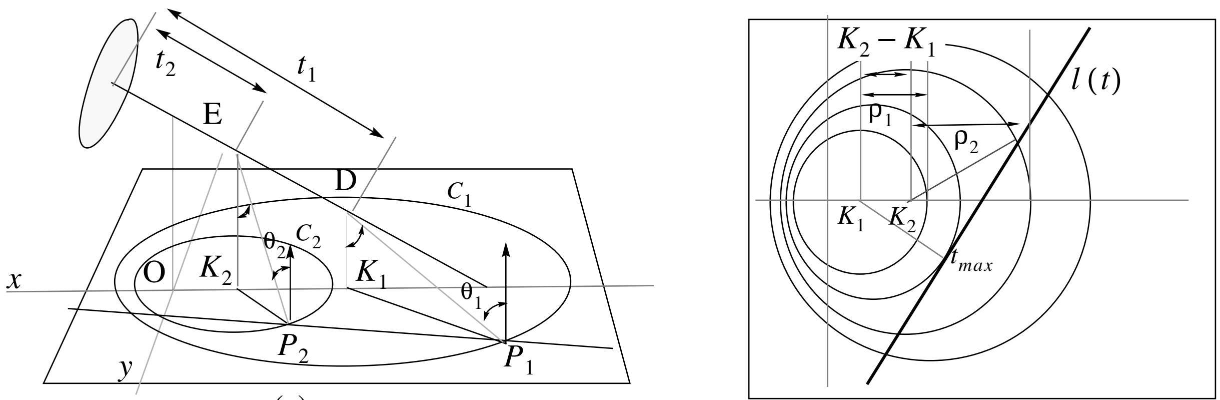
$$R_s(t) = \sqrt{t^2 + a^2} \tag{3-2}$$

Therefore the radius of the resulting isolux sphere is an increasing function of t. From Eq. (3-1), it is known that the radiance value of the isolux sphere is a decreasing function of R_s, and thus a decreasing function of t. Consequently, of all the isolux spheres that have at least one point in common with the receiving plane, the sphere $S(t_{min})$ has the smallest radius, and consequently the highest value of illumination on the receiver plane. Therefore the function $L(x, y)$ has a single maximum at the point which $S(t_{min})$ touches the plane $z = 0$. □

The origin O is now placed at P_{max} and the x-axis aligned with the projection of the line L (see Figure 3-4(c)). To complete the proof for the first term, it suffices to show the following.

Lemma 3-2: *The restriction of $L(x, y)$ on any line on the plane has a single maximum t_{max} and is decreasing as a function of distance from t_{max}.*

To show Lemma 3-2, a geometric argument is used. Consider the geometry shown in Figure 3-5(a). Let K_i be the x-coordinate of the projection of the centre of the sphere $S(t_i)$ onto the x-axis. For each sphere there corresponds a circle, say C_i, with radius ρ_i, which is the intersection of the



(a) (b)

Figure 3-5. Proof of Lemma 2

sphere with the plane. It can be shown, by examining the geometry, that if $t_j > t_i$ and consequently $K_j > K_i$, then $\rho_j > \rho_i$, and $\rho_j > \rho_i + (K_j - K_i)$. This can be seen in Figure 3-5(b) for K_1 and K_2. Therefore on the plane there is a set of circles that have centres K_i along the *x*-axis, and for $K_j > K_i$, circle C_j entirely encloses C_i. This is shown in Figure 3-5(b).

For any line $l(t)$ crossing this family of circles, there will be only one circle that is tangential to the line, and this point will correspond to the isolux sphere of smallest radius, and therefore the point of maximum illumination t_{max} (see Figure 3-5(b)). The circles intersected as a point moves away from the maximum along the line will have ever increasing radii, and therefore correspond to decreasing illumination values.□

Given Lemma 3-1 and Lemma 3-2, it has been shown that the first term of the radiance from a disc source is unimodal. It is now shown that the cosine term preserves unimodality.

Lemma 3-3: *The cosine scale factor* $\cos\theta$ *decreases with distance from the maximum.*

Proof: Referring to Figure 3-5(a), the angle in question for P_1, is: $\theta_1 = \angle K_1 D P_1$ and for P_2: $\theta_2 = \angle K_2 E P_2$, since D and E are the centres of the corresponding isolux spheres. Since it is assumed that $K_1 > K_2$ and that $\rho_1 = K_1 P_1 > \rho_2 = K_2 P_2$, it can be seen that:

$$\theta_1 > \theta_2 \Leftrightarrow \cos(\theta_1) < \cos(\theta_2) \qquad (3\text{-}3)$$

since $0 < \theta_1, \theta_2 < \pi/2$, and therefore the scale factor preserves the nature of the illumination function that decreases with the distance from the maximum.□

With the proof of Lemma 3-3, it has been shown that the function of illumination due to a disc source on a plane with arbitrary orientation is unimodal.

3.1.1.2 Unimodality of Radiance Functions due to Convex Polygonal Light Sources

Proving unimodality for general light source configurations is a difficult task. It is possible to show unimodality for restricted geometries. In Appendix A.2, it is shown that radiance from a rectangular source on a receiver parallel to the plane of the source is unimodal. This is shown using a simple antiderivative argument. Given the proof for the disc source and that for the restricted rectangular case, the following conjecture is proposed.

Conjecture 3-1: *The radiance functions of diffuse planar convex polygonal light sources on arbitrarily oriented, diffusely reflecting receiver planes, are unimodal.*

The general proof for the disc light source gives us strong evidence that this property holds, since it can be seen that moderately tight bounds can be constructed from disc light sources for a number of polygonal shapes. Finally, in all the experiments performed in this research, the illumination functions on surfaces due to convex sources have demonstrated unimodal behaviour. A proof of the conjecture is a subject of current research.

3.1.2 Derivatives and Curvature

For simplicity, in the following discussion a one-dimensional cross section of the full radiance function $L(x, y)$ is considered, defined as $I(t)$, with the parameter t varying along a line. It is important to note that this analysis only gives a rough idea of how the two-dimensional function behaves.

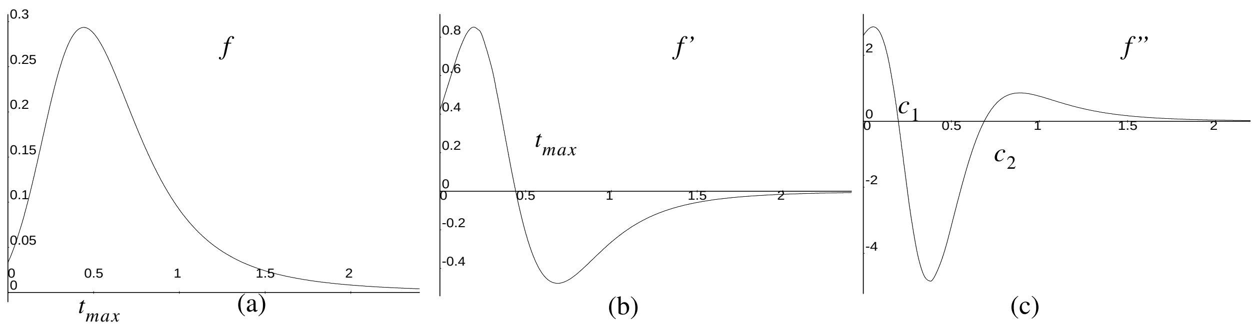


Figure 3-6. A radiance function and its first and second derivatives

What can be concluded about a function's behaviour given unimodality, and continuity of the first and second derivative? For a given function $I(t)$, it is immediately known that within a relevant domain the first derivative has one root, at the maximum. Determining the position of this maximum is important. Let t_{max} be the value of t such that $I(t)$ is maximum. Due to the decreasing behaviour of the function with respect to the distance from t_{max}, the derivative appears in practice to have one maximum and one minimum. A formal proof showing that there is only one maximum and only one minimum is difficult, even for the disc case. For a polygonal source, it would imply the proof of Conjecture 3-1.

If the existence of exactly one maximum and exactly one minimum is assumed for the first derivative, the second derivative will have exactly two roots. It must be noted that the derivatives for example in Eq. (3-1), or (2-26) become large very quickly. The radiance function originally shown in Figure 3-2(b) is repeated in Figure 3-6(a), its first derivative is shown in Figure 3-6(b), and the second derivative in Figure 3-6(c).

One important measure of a function's smoothness is curvature. The curvature of a function f is defined as follows:

$$\kappa(t) = \frac{f''}{[1 + (f')^2]^{3/2}} \tag{3-4}$$

When curvature changes sign, the function changes from convex to concave. To determine these crossings, called inflection points, it is sufficient to find the roots of the second derivative. We shall call the inflection points c_1 and c_2, together with t_{max}, the *critical points* of the function. The critical points of a typical radiance function f are shown in Figure 3-6. For the two dimensional function $L(x, y)$ the inflection points c_1 and c_2 correspond to a *critical curve*, that is typically of high degree in *x* and *y.*

3.2 Piecewise Polynomial Radiance Interpolants

Our goal is to develop an algorithm that will achieve an *accurate* and *efficient* polynomial representation of radiance over a surface. It is thus desirable to use low degree polynomials, and to incorporate tight error bounds into the algorithm.

In the regions of the receiver outside the critical curve, radiance tapers off asymptotically and monotonically to zero. As a result, while the regions close to the maximum are well behaved, the regions further away cannot easily be approximated with polynomial interpolants of degree greater than one, without introducing artifacts. This classification carries over to the restriction of the radiance function to a line in which four regions are distinguished: $\{ [t_s, c_1], [c_1, t_{max}], [t_{max}, c_2], [c_2, t_e] \}$. The regions $[t_s, c_1]$ and $[c_2, t_e]$ are called the *tails* of the illumination function. Within each one of these regions, radiance is either strictly increasing or decreasing, and either concave or convex.

By performing this segmentation and guaranteeing monotonicity properties in each region, it is possible to avoid undersampling and ringing artifacts in polynomial interpolation such as undershooting and overshooting. The region inside the critical curve around the maximum is perceptually important since it is the brightest region of the surface. By identifying the maximum, special care can be taken to achieve high-quality interpolation in the region around it.

Before describing how these observations lead to an algorithm for sample placement and reconstruction, it is necessary to describe certain issues relevant to the representation of radiance by polynomial interpolants.

3.2.1 Representing Illumination with Piecewise Polynomial Interpolants

To maintain the original goal of efficiency, the reconstruction scheme should require minimal computation to determine a value of the desired function. Low order polynomial interpolants have such characteristics, since they require a small number of multiplications to evaluate a function at any point. Following these principles, we limit ourselves to linear and quadratic interpolants.

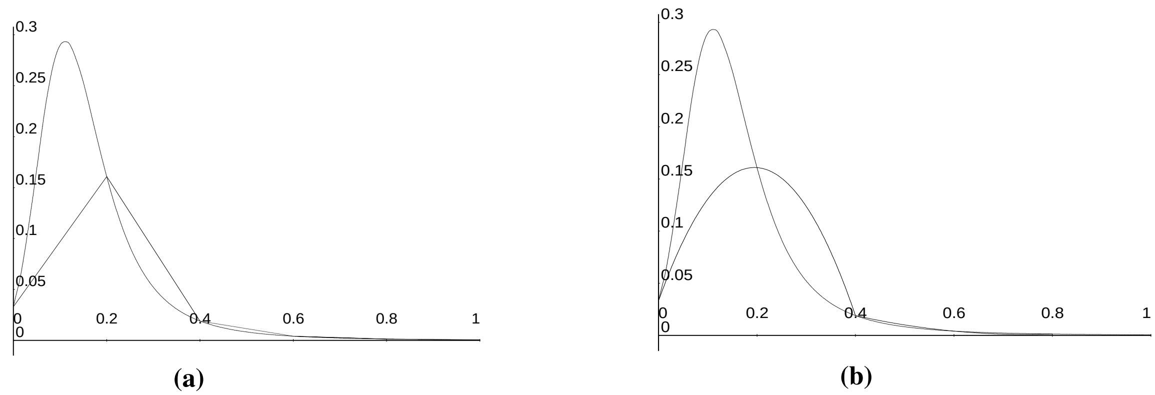


Figure 3-7. Undersampling Problems from Uniform Linear and Quadratic Interpolation

Higher order interpolants can introduce ringing artifacts, and therefore should be used with caution. As discussed in Section 2.3, cubic [49] and quadratic [38] interpolants have been used in previous work. To see why caution must be used in these cases, let us consider a specific example of uniform linear and quadratic interpolation. The function originally depicted in Figure 3-2(a) is used, over a receiver that is twice as long to the right, to demonstrate the problems more clearly. Using 5 samples, the uniform quadratic and uniform linear interpolation of this function are shown in Figure 3-7(a) and (b), respectively. While one may argue that we are deliberately undersampling, we remind the reader that our intent is to reduce the number of evaluations of the radiance function.

As can be seen in Figure 3-7, when the function structure is not taken into account, significant errors can result. The maximum value can be missed, as is the case in both linear and quadratic approaches. This can be disastrous, since in the "radiosity gradient" approaches, such initial undersampling will mean that the elements may not be further subdivided, and the maximum illumination will be drastically underestimated. By identifying the maximum t_{max}, this artifact can be completely avoided.

Another interesting artifact is that of over- and under-shooting caused by naïve quadratic interpolation (shown in Figure 3-8). Even though negative lobes can be immediately identified without resorting to the original function structure, non-negative artifacts cannot be consistently identified.

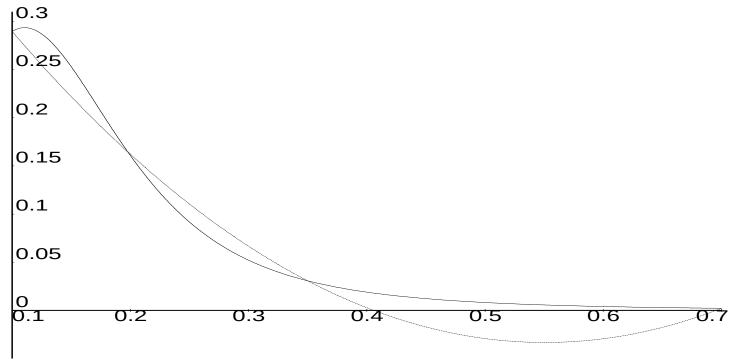


Figure 3-8. Over- and Under-Shooting due to a Quadratic Interpolant

It is only possible to identify such artifacts if the regions being examined are guaranteed to be either concave or convex. This can only be accomplished by identifying the critical points.

Rendering

The polynomial representations of radiance are used for shading objects illuminated by area sources. The interpolants can be queried in a ray-casting renderer (this approach is also used to produce the images by Lischinski et al. [38]) or a z-buffer system to extract high quality values on a pixel by pixel basis (see Figure 3-9(a)). The interpolants can also be resampled at the vertices of a regular polygonal grid to be subsequently used in hardware rendering (see Figure 3-9(b)).

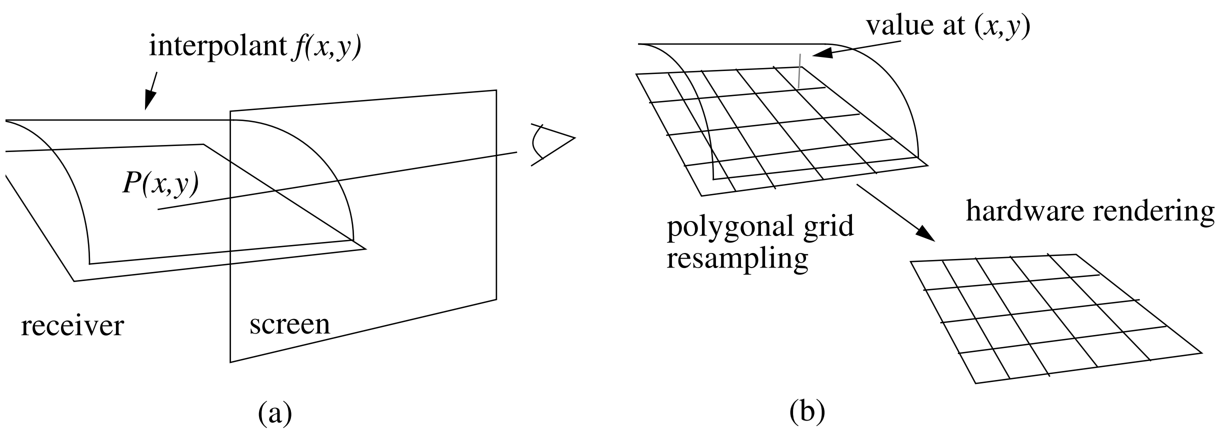


Figure 3-9. Ray Casting and Hardware Rendering

3.3 The Structured Sampling and Reconstruction Algorithm

From the discussion in the previous sections, it is clear that significant properties of radiance function from diffuse area sources can be exploited. The functions can be assumed to be smooth, unimodal, and it is important to segment them into regions with convex or concave behaviour. This knowledge is a powerful tool that has henceforth not been used to develop sampling strategies for radiance reconstruction. In the remaining sections of this chapter an algorithm that takes advantage of these properties is developed, and results of an implementation are presented.

Initially the maximum is found, if it is within the receiver surface. Exactly determining the curve of zero curvature would be a difficult task, so the approach taken is find the inflection points on two orthogonal axes that pass through the centre of the receiver surface (see Figure 3-10(b)-(c)). Adaptive subdivision is performed along these two axes, producing two lists of one-dimensional segments, on which a linear or quadratic interpolant is fitted (see Figure 3-13(b)). These lists are then used to create a two dimensional interpolant (e.g., a tensor product shown in Figure 3-13(c)). The algorithm can be summarised in the following pseudocode segment.

```
sample_and_build_interpolant( Source, Receiver_Polygon )
{
    { list_x, list_y } = find_max_and_create_lists()
    for list in { list_x, list_y } do
          find_inflection_points( list )
          adaptive_subdivision( list )
    done
    build_2d_interpolant( list_x, list_y )
}
```

The call to `find_max_and_create_lists()`, serves a double purpose. The maximum is found, if it exists, and the two lists are initialised to contain the starting points, x_s, y_s the endpoints x_e, y_e, and, if applicable x_{max}, y_{max}. The following sections detail each of the steps of the algorithm.

3.4 One Dimensional Analysis for Subdivision

As mentioned above, it is much easier to deal with the restriction of radiance on two orthogonal lines than with the full two-dimensional function. In particular, the maximum can be found using a two-step one dimensional procedure. In addition, instead of determining the critical curve, the inflection points are found on the two axes. The construction of the final two-dimensional interpolant is described in Section 3.5.

3.4.1 Finding the Maximum and Approximating the Curve of Zero Curvature

Since the radiance function is assumed to be unimodal, a good approximation to the mode can be found. We first find the maximum x_{max} along a line parallel to the *x*-axis, then we find the maximum y_{max} along a line l_y parallel to the *y*-axis, that passes through x_{max}. The point (x_{max}, y_{max}) is an approximation to the overall maximum. When the source polygon has edges with a high aspect ratio, it may be necessary to perform a subsequent iteration to find a better approximation. The quality of the approximation is checked by seeing if the derivatives on the two diagonals to the axes are zero at the maximum.

In the routine `find_max_and_create_lists()` a bounding box is placed around the receiver surface, one edge of which is aligned with the longest receiver edge. The two-dimensional local coordinate system is defined as this edge being the *x*-axis, while the first vertex of the bounding box defines the *y*-axis. The midpoints of the lines $x = 0$ and $x = x_e$ are found and one-dimensional analysis is performed on the line defined by these two points (see Figure 3-10(b)). It is first determined whether a maximum exists along this line, and if there is one, it is found numerically. If there is a maximum x_{max}, the line passing through $(x_{max}, 0)$ that is parallel to the *y*-axis is formed, and the maximum y_{max} along this line is found (see Figure 3-10(c)). The point (x_{max}, y_{max}) is the overall maximum, if it is contained in the original receiver bounding box. Two lists are now constructed, `list_x` and `list_y`. If the maximum is contained in both, `list_x` = $\{x_s, x_{max}, x_e\}$ and `list_y` = $\{y_s, y_{max}, y_e\}$, else they contain just the endpoints (Figure 3-10(e)). If the maximum is not contained in the polygon, the routine searches for any remaining critical points along the axes.

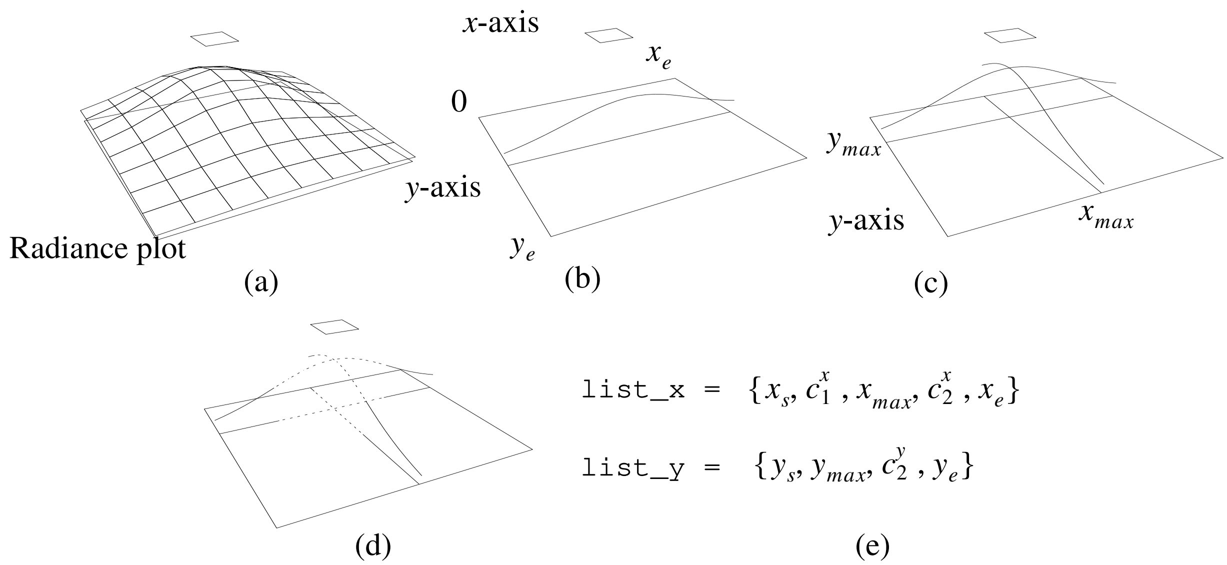


Figure 3-10. Finding the Maximum and the Critical Points.

The next step is to find the inflection points along each of the lines parallel to the *x*-axis and the *y*-axis respectively. Each of the two lists is processed independently by the routine mentioned above, `find_inflection_points()`. The points c_1, c_2 are found for each list, if they are contained between the endpoints. There are now two lists segmented into regions defined by the maximum and the inflection points (see Figure 3-10(e)). This segmentation thus guarantees that the function is monotonic, and either convex or concave in each of these segments (see Figure 3-10(d)). These segments are passed to the adaptive subdivision routine, described in the following section.

3.4.2 Adaptive Subdivision

To maintain the original goals of efficiency and accuracy, a consistent subdivision criterion is required. This allows the adaptive subdivision of `list_x` and `list_y` so that the interpolants fitting in each subdivided region or interval are guaranteed to meet a predefined tolerance or error bound. Subdivision is thus based on determining an upper bound on the error after a specific subdivision step, and on using this bound to determine if subsequent subdivision is necessary. The termination criterion is given in terms of the absolute error tolerance in pixel values (i.e., pixels that differ by less than `tol` on a [0,...,255] RGB value scale) the user is willing to tolerate between the approximation and the exact solution.

The adaptive method employed is guided by the following general principle. Given a one-dimensional region of the parameter t, try to fit a parabola to the function in that region. If an acceptable error bound can easily be determined, stop. If not, attempt to fit a linear approximation within an acceptable error bound, and if that also fails, subdivide until a maximum subdivision level is reached. It must be emphasized here that all function values previously computed are reused in the adaptive subdivision steps, and are also used for the final interpolant construction.

```
adaptive_subdivision( list )
{
    seg_left = list_start;
    if( list contains c1 )
         seg_left = subdivide_tail( seg_left, c1 );
    if( list contains tmax ) {
         seg_left = subdivide_c1c2_region( seg_left, tmax );
         if( list contains c2 )
              seg_left = subdivide_c1c2_region( seg_left, c2 );
         else
              seg_left = subdivide_c1c2_region( seg_left, list_end );
    }
    if( seg_left != list_end )
         subdivide_tail( seg_left, list_end );
}
```

Figure 3-11. Pseudocode for Adaptive Subdivision

The segments contained in each list are either in a tail region, or in the region contained by the inflection points $[c_1, c_2]$, in each of the x and y directions. The adaptive subdivision algorithm is summarised in the pseudocode segment of Figure 3-11.

The list is examined to see if the maximum and the inflection points are contained, and the appropriate subdivision routines are called. A trivial reject test is performed first, to determine if the overall variation along the line allows the representation of the function by a single segment.

3.4.2.1 Adaptive Subdivision in the Region $[c_1, c_2]$

The behaviour of the radiance function is well known in the regions $[c_1, c_2]$ around the maximum, since their size does not depend on the size of the receiver, but only on the distance and orientation from the source. Thus, radiance can often be quite accurately approximated by a cubic. If this is the case, a quadratic is chosen that minimises the error between the cubic and the quadratic. If this is not possible the algorithm for the tails is applied.

The algorithm initially is given a region $[a, b]$ which is a subset of either $[c_1, t_{max}]$ or $[t_{max}, c_2]$. Initially a parabolic fit to the function in this region is attempted. To achieve this a third point is required. Let $Q_u(t)$ be the parabola that interpolates the endpoints of I at $(a, I(a))$, $(b, I(b))$ and an variable third point $(u, I(u))$. Ideally u should be chosen such that $\sqrt{\int_a^b ((Q_u(t) - I(t)))^2 dt}$ is minimum. Finding this value of u is too expensive, since the integral would have to be numerically evaluated at significant cost.

As mentioned above, in this region, it is possible to closely approximate the function in one dimension using cubic hermite curves [1]. Using the hermite interpolants allows the direct solution of the following minimisation problem. Define $C(t)$ to be the hermite cubic approximating the function, and let $Q_u(t)$ be the quadratic defined by $(a, I(a))$, $(b, I(b))$ and $(u, C(u))$. Find u such that

$$F(u) = \sqrt{\int_a^b (Q_u(t) - C(t))^2 dt} \tag{3-5}$$

is minimum. Call the value of u such that $F(u)$ is minimum u_m. If Q_{u_m} is acceptable (see below), then an error bound for this parabola is can be determined by finding the maximum value of:

$$B = \left|Q_{u_m}(t) - C(t)\right|. \tag{3-6}$$

This maximum can be computed directly, since the coefficients for both the parabola and the cubic are known.

The parabolic fit is unacceptable if its maximum is in the interval $[a, b]$. This test can be performed trivially. If this is the case, u_m is set to $(a + b) / 2$ and the parabola Q_{u_m} is used. If the

1. The final two-dimensional interpolant is linear or quadratic. The cubics are used only in the one-dimensional subdivision process. The quality of approximation achieved appears to be very good in practice.

parabola is acceptable, the value B is tested, and if it is within tolerance the algorithm stops. If not, a linear fit is attempted, using the previously determined point u_m to generate two linear segments, L_1 and L_2. If this fails, the interval is subdivided.

To bound the error incurred by the linear segment L_i, we find the maximum of the function $|L_i(t) - C(t)|$ in the interval, an operation that again can be performed analytically. The value of t for which this is maximum is called t_m. In this case the error in the interval due to the linear approximation is bounded by:

$$B = |L(t_m) - C(t_m)| \tag{3-7}$$

For each of the two segments B is compared to the tolerance. If it is less, the algorithm stops, else the interval is subdivided.

3.4.2.2 A Midpoint Tail Subdivision Scheme

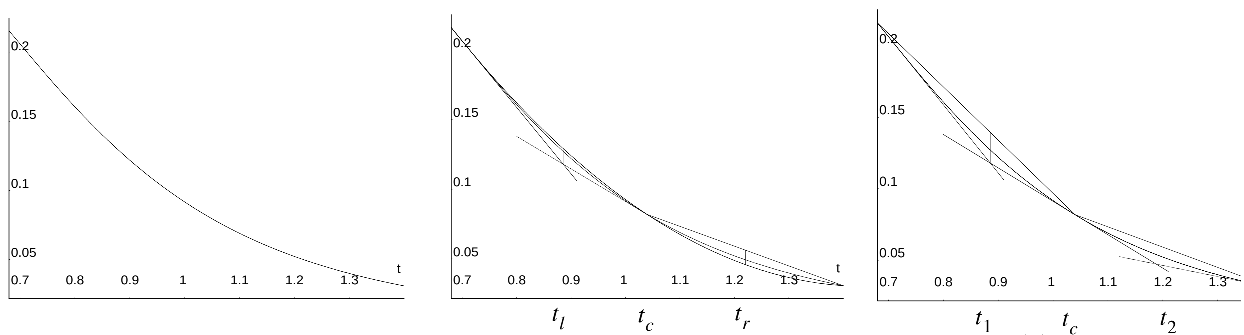
Due to the asymptotically diminishing nature of radiance within the tail regions, cubic approximation invariably leads to ringing artifacts. This problem becomes particularly acute when the receiver is very large compared to the source. Thus a midpoint subdivision algorithm is employed in these regions. Tight error bounds are determined based on the derivative. Examples of the error estimates are shown as vertical lines in Figure 3-12(b) and (c).

As an alternative to cubics, a midpoint subdivision approach is employed. The algorithm proceeds as follows.

1. Assume that the original interval for subdivision is $[a, b]$. Create the parabola defined by $(a, I(a))$, $(b, I(b))$ and $((a+b)/2, \quad I((a+b)/2))$.
2. Determine if the parabola is an acceptable fit. If so determine the error bounds in both subintervals due to the parabolic interpolation. If both error bounds are under the acceptable limit stop. If not, proceed to the following step.
3. Determine if the errors from linear interpolants are acceptable in both subintervals. If they are, stop. If one regions' bound is above tolerance subdivide and leave the other as is. If not, subdivide and perform Step 1 for both subintervals.

Because the regions have been subdivided so that the function in $[a, b]$ is either convex or concave (see for example Figure 3-12(a)), any quadratic whose minimum/maximum lies well within the interval $[a, b]$ can be immediately rejected. If this is not the case the quadratic crosses the function $I(t)$ once in $[a, b]$. Call this crossing point t_c (see Figure 3-12(b)).

The interval $[a, b]$ is now split into two, and it is known with certainty that the parabola is completely above or completely below the illumination function in either subregion (see Figure 3-12(b)). Using this fact, the interval in which the parabola is entirely above the function can immediately be determined, and so can the interval in which it is below, by examining the derivatives of the quadratic and the function at a and b.



(a) (b) (c)

Figure 3-12. Bounding the Error in the Tail Regions

For the subregion of the parabola that is below the radiance function, say $[t_c, b]$ without loss of generality, the point can be computed at which the parabola is furthest from the line segment $L(t)$ defined by $(t_c, I(t_c))$ and $(b, I(b))$. This situation is shown on the right hand side of Figure 3-12(b). Call this point t_r. An upper bound on the error incurred by the parabola in the subinterval $[t_c, b]$, is

$$B = |Q(t_r) - L(t_r)| , \tag{3-8}$$

expressing the maximum distance of the quadratic to the line $L(t)$. The vertical line in the right half of Figure 3-12(b) shows the magnitude of B.

For the subregion that is entirely above the parabola, the method used for linear interpolants to find t_l described below is used, but the bound is taken to be:

$$B = |Q(t_l) - L(t_l)| \ . \tag{3-9}$$

The vertical line in the left half of Figure 3-10(b) shows the magnitude of B.

Bounding the Error for a Linear Interpolant

To bound the error incurred by the line segment L defined by $(a, I(a))$, $(b, I(b))$, the derivatives $I'(a)$ and $I'(b)$ are used. Since it is known that the function is either convex or concave in this interval, two lines l_a and l_b can be created that have slope determined by $I'(a)$ and $I'(b)$, that will have one intersection in the interval $[a, b]$. These lines, for both left and right hand subintervals, are shown in Figure 3-12(c). Call this point of intersection (t_m, I_m). The error in $[a, b]$ is bound by the value:

$$B = |I_m - L(t_m)| . \tag{3-10}$$

In Figure 3-12(c), the error bound is shown by a small vertical line for both intervals, where t_m corresponds to t_1 and t_2 respectively. This bound is quite tight in most cases, as shown in the example of Figure 3-12(c).

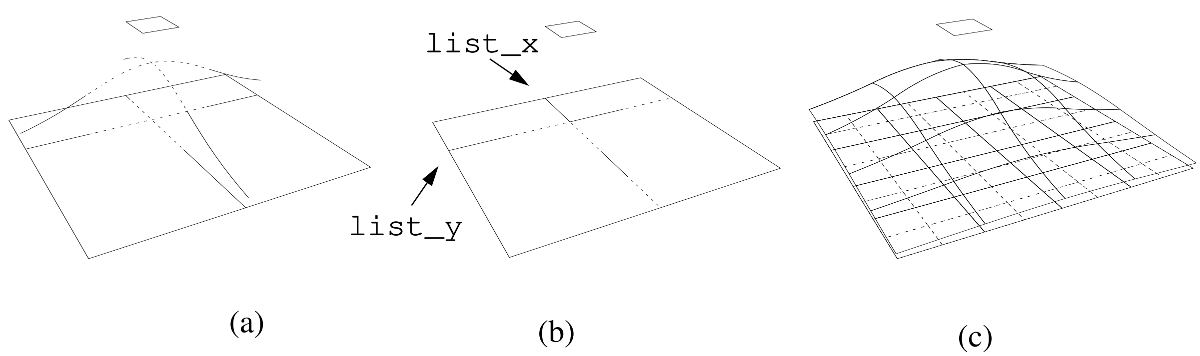


Figure 3-13. Building the Two-Dimensional Interpolant

3.5 Building the Two Dimensional Interpolant

After the adaptive subdivision of both `list_x` and `list_y` the remaining task is the construction of the two-dimensional interpolant for radiance. The two lists (Figure 3-13(b)), can be used to construct two-dimensional interpolants. Two different approaches have been implemented, a tensor product construction and a triangular grid construction.

Throughout this thesis, interpolating forms are used for approximation, specifically basis functions of the Lagrange family. Interpolation is desirable in the context of illumination approximation, since accurate representation of the function values tends to be more important than smoothness, that could be achieved using other non-interpolating bases such as B-splines. In terms of representing radiance over a surface, it seems more intuitive to have an approximation for which the exact values exist at the nodes of the mesh of sample points over which the interpolant is constructed.

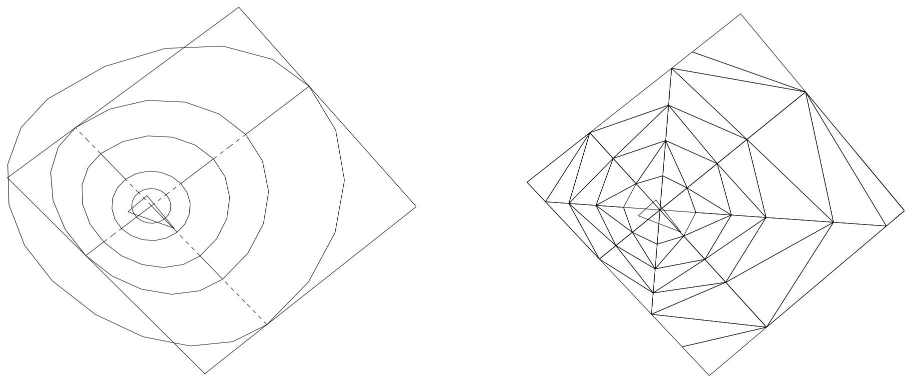
3.5.1 Tensor Product Construction

For the tensor product construction, each list is traversed, and the corresponding one-dimensional segment is used as the x or y side respectively of the two-dimensional interpolant. This results in two-dimensional tensor product Lagrange interpolants. If both lists correspond to quadratic interpolants, the two-dimensional interpolant will be a biquadratic surface. Otherwise the interpolants will be mixed linear-quadratic surfaces, or bilinear surfaces if both sides are linear. An example tensor product interpolant is shown in Figure 3-13(c).

3.5.2 Triangular Grid Construction

Given the one-dimensional analysis and the subdivision of the tails, interpolants over triangular elements can be used instead of the tensor product formulation described above. The triangular formulation capitalises on the unimodal nature of the radiance function to further reduce the computational expense.

The adaptive subdivision along the two axes in the receiver plane defines four quadrants. On each axis a list of segments exists, which correspond to approximately the same changes in illumination, since the same adaptive subdivision algorithm has been applied to each axis. The segments that change from solid to dashed in Figure 3-14(a) show this subdivision. A curve passing through the vertices of this subdivision on each of the axes will therefore approximate an isolux contour. To approximate this curve, a quadrant of an ellipse is formed using the corresponding vertices on each axis as major and minor ellipse axes. The ellipses are shown in Figure 3-14(a). Diagonals are then drawn across each quadrant, and a triangular mesh is built based on this subdivision (Figure 3-14(b)).



(a) **(b)**

Figure 3-14. "Isolux" Ellipse Sampling and the Corresponding Triangulation

In this approach, it is not necessary to evaluate the function along the diagonals since the vertices on the diagonals are on isolux contours, and therefore have approximately the same value as the vertices of the generating axes. Also, the error bounds found during the adaptive subdivision are much more closely related to the interpolants along the diagonal directions. Therefore better confidence can be had in the quality of the two-dimensional approximation to the function.

The drawback of this approach is that triangular elements tend to result in visual artifacts across the diagonals. There are two possible solutions. The first is to use quadratic interpolants, which was the approach adopted in the first implementation. A midpoint quadratic interpolant is applied if the error bounds are maintained. To determine the error the quadratic interpolant bounds check described in Section 3.4.2 is used. To maintain continuity it is necessary to create a special mixed quadratic/linear triangular element, by using a basis function that is quadratic on one edge and linear on the other two. The second solution involves finer angular subdivision, by subdividing the regions between the axes and the diagonals. This is a low cost solution, since again the function values can be extracted from the isolux contour, but requires more complex data structures.

3.6 Implementation and Numerical Issues

The above sampling and reconstruction algorithm was implemented in a global illumination test-bed. The environment is built in C++, including a set of classes that implement sampling and reconstruction.

3.6.1 Numerical Issues

To find the maximum and the critical points, Brent's minimisation algorithm was used [36]. For a given interval $[a, b]$, it is first determined whether t_{max} is in the interval, by examining the values of the derivative at a and b. If the maximum exists, it is found by maximising the illumination function. It must be noted that this expense can be avoided when the maximum can be found geometrically, as is the case for the disc source. In a similar fashion, the points c_1 and c_2 are found, if necessary, by minimising or maximising the first derivative. The average expense of this iterative procedure over a large number of tests is between 4 and 6 iterations.

The illumination functions due to polygonal sources (Eq. (2-26)) have complicated and unstable derivatives. Finite differences are used to approximate the derivative values. It is important however to have good derivative values for the endpoints used in the cubic hermite analysis of Section 3.4.2.1. A more stable, albeit more expensive, derivative calculation is thus used at these points, namely Ridders' method [45].

As discussed above the subdivision algorithm will generate a list of segments in each of which the error will be bounded by the user-specified tolerance. However, numerical instabilities can create very small intervals, and cause endless subdivision. When a very small interval or subinterval is detected, the subdivision stops.

One further problem with this approach is that discontinuities of the first derivatives can become evident visually, even though the error tolerance has been met. This is particularly evident when a linear edge (of a bilinear or quadratic-linear tensor product) meets a quadratic edge. It is thus beneficial to bias the algorithm towards choosing quadratic segments even if a completely strict analysis would require the rejection of a quadratic interpolant.

Specifically, when attempting the parabola fit to the function using the steps described in Section 3.4.2.1 or 3.4.2.2, the parabola is not rejected directly if overshooting or undershooting occurs. Instead, the area contained between the derivative and the portion of the parabola over- or under-shooting is calculated. If this is very small, the ringing is ignored.

The algorithm has to be modified slightly to deal with the case where the source and the receiver share a common edge, since the derivative of radiance has a singularity. As described above, the error estimates depend on the derivative, and are thus unstable in these cases. When such a common edge is detected, we force additional subdivision close to the common edge, and use linear interpolation. The difference of the radiance values of the endpoints is used as a conservative error bound for the linear interpolant.

3.6.2 Successive Parabolic Interpolation Algorithm to Find the Inflection Points

One of the major expenses of the structured algorithm in its first implementation is the determination of the inflection points. The inflection points are found, as mentioned above, by Brent's minimisation algorithm applied to the derivative. Since the evaluation of a derivative requires between 2 and 4 evaluations of the original functions, even 4-6 iterations can result in about 20 function evaluations. A different way of looking at this is that a competing uniform method can start with a 5 by 5 grid before the structured method has had a chance to use one sample towards display.

As an example of how this initial cost can be significantly reduced, we present a successive parabolic interpolation algorithm to find the inflection points. The algorithm proceeds as follows.

First a list is constructed containing all the points evaluated during the minimisation process to determine the maximum. These are evaluations of the one-dimensional function $I(t)$ in the original interval $[t_s, t_e]$. This list is maintained sorted with respect to the parameter *t*. The algorithm splits the interval into two, namely $[t_s, t_{max}]$ and $[t_{max}, t_e]$. It is assumed that there are at least four points (including the endpoints), in each of the subintervals. If not, more evaluations are performed. We denote the sequence of points in the interval for which $I(x_i)$ has already been evaluated, $(x_1, \ldots, x_n)$.

The algorithm examines every quadruple $(x_i, x_{i+1}, x_{i+2}, x_{i+3})$, fitting the quadratic segments Q_1, Q_2 defined by the triples of points:

$$(x_i, I(x_i)),\ (x_{i+1}, I(x_{i+1})),\ (x_{i+2}, I(x_{i+2})) \tag{3-11}$$

$$(x_{i+1}, I(x_{i+1})),\ (x_{i+2}, I(x_{i+2})),\ (x_{i+3}, I(x_{i+3})) \tag{3-12}$$

When a quadruple is found so that Q_1 is convex and Q_2 concave or vice versa, it is known that the inflection point is with the interval $[x_i, x_{i+3}]$. Concavity is tested by examining the sign of the first coefficient of the parabola. If there is no such interval, the two internal points that are most evenly spaced are taken to create a starting quadruple.

Call the points of the starting interval (x_0, x_1, x_2, x_3), which bracket the inflection point. Fit the quadratic segments Q_1 and Q_2 as defined by Eq. (3-11) and (3-12) respectively. One of the following three situations can occur.

1. Q_1 and Q_2 are both concave. In this case the inflection point is between x_0 and x_1.
2. Q_1 and Q_2 are both convex. Then the inflection point is between x_2 and x_3.
3. Otherwise the inflection point is between x_1 and x_2.

In each case two new points are taken in the interval containing the inflection point, and this interval with the two new points becomes the quadruple under consideration. The algorithm stops when the interval containing the inflection point is smaller than some given tolerance.

3.7 Results

The algorithm that constructs tensor products has been implemented and tests run over a suite of simple environments. A preliminary test of the triangular approach has also been performed along similar lines.

3.7.1 Error Metrics and Comparison Conditions

Evaluating the quality of the new method is a difficult task. All error calculations use an analytically computed solution of the image in question, generated using the formula of Eq. (2-26) for a polygonal light source, to compute the exact radiance at any pixel.

A suite of error metrics is used. First a pixel-by-pixel image difference from the analytic solution was used (a similar metric was used in [19] and [20]). The ratio of the number of pixels that display more than `tol%` difference in absolute value from the analytically generated image over the total number of visible pixels gives a percent error metric. This first metric gives a rough idea of how many pixels display a large difference from the exact image.

A second metric is a root-mean-squared metric, defined as follows for an image of size *xRes* by *yRes*:

$$RMS = \sqrt{\sum_{x=0}^{xRes}\sum_{y=0}^{yRes}\frac{(P(x,y) - P_a(x,y))^2}{(P_a(x,y))^2}} \qquad (3\text{-}13)$$

where $P(x, y)$ is the value of radiance for the pixel (x, y) computed by the approximation algorithm in question and $P_a(x, y)$ is the exact (analytic) value of radiance for the pixel. Finally the average and maximum pixel error (in terms of RGB values in the interval [0,...,255]) for a sequence of images are also reported.

For each test sequence two comparisons are made. The first is with tensor product interpolants on uniform grids that contain the same number of samples as those in the final tensor product grid output from the structured sampling approach.

In the second comparison, compensation is made for the expense incurred in finding the critical points and adaptive subdivision. This is achieved by counting the number of function evaluations incurred in the sample placement and the adaptive subdivision. To achieve “fair” comparison to a uniform solution, a uniform grid is constructed that has the same number of sample points as the structured solution, plus as many additional sample points (within integer limitations) as those used in the sample placement stage. For example, if sample placement cost 25 function evaluations, and the structured grid is 8 by 7, the uniform solution will have a 9 by 9 grid. This is justified as follows: the total number of function evaluations used to compute the structured solution was 25 + 56 = 81, and the total number of function evaluations used for the uniform solution must be the same, 9 x 9 = 81. Both uniform linear and uniform quadratic interpolants are compared to the structured solution.

The additional function evaluations are required to find the maximum and the inflection points, and also during the adaptive subdivision steps when derivative values are calculated. It must be emphasized that this "fair" comparison is thus completely dependent on the methods and implementation used for these steps. The times reported in this section are from the first implementation of the algorithm. It is expected that with the use of methods such as successive parabolic interpolation, and geometric approximations for the maximum and inflection points, this additional expense will be significantly lower, making the first comparison even more relevant.

3.7.2 Test Results for Structured Tensor Product Interpolants

Two sequences of 20 frames are presented as a test suite. The first involves four walls of a room and a floor, in which a rectangular light source moves across the room. The grid resulting from the structured solution for Frames 2 and 13 is shown in Figure 3-15. The second test suite is of the same room, in which the light source rotates around the horizontal axis. Frames 2 and 13 are shown in Figure 3-16. The dashed lines correspond to the midpoints of quadratic segments, while solid lines are the boundaries between tensor product elements.

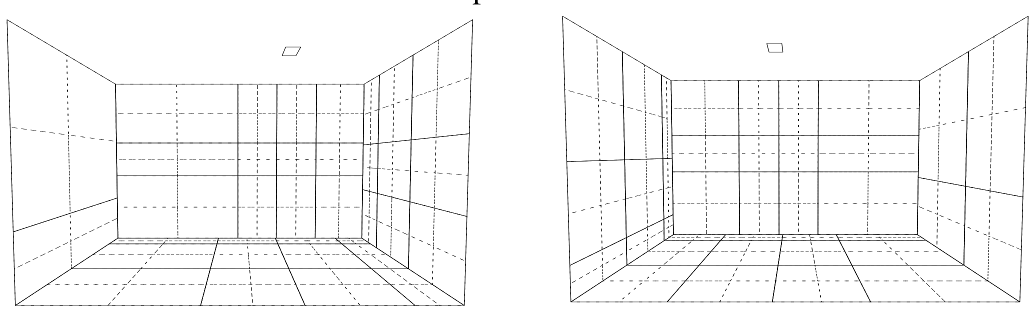


Figure 3-15. Frames 2 & 3 of the Test Suite LS Translation (Shaded Structured Solution)

Notice how the grid adapts to the different position of the light source as it moves across the room or rotates around an axis, assigning more samples where they are needed, and fewer when the power and variation drop. Recall that each interpolant segment is guaranteed to satisfy a preset user tolerance of pixel difference from the analytic solution.

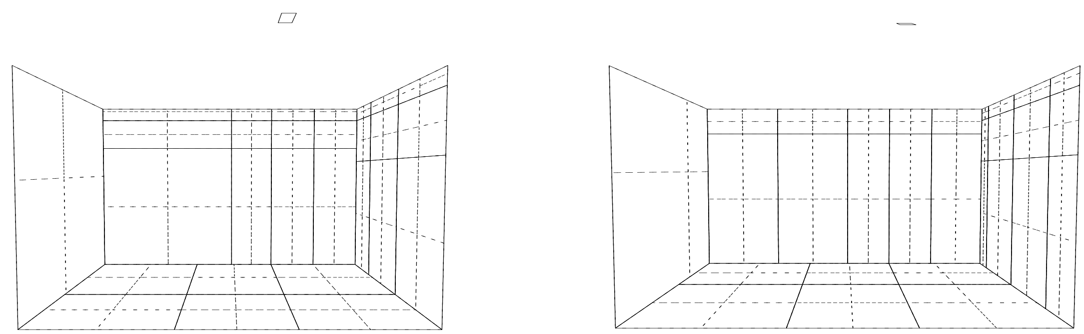


Figure 3-16. Frames 2 and 13 of Test Suite with Light Source Rotation (Structured Grid)

The average values over twenty frames of animation, for the error metrics discussed in the previous section are shown in Table 3-1 for the structured and the uniform quadratic solutions for source translation, and in Table 3-2 for source rotation. In addition the total number of samples used in the interpolants over the twenty frames is reported, as well as the number of samples for the structured solution. The same error metrics are shown for the uniform quadratic and uniform solution when compensation has been made for the additional expense incurred by the structured algorithm.

	Average% Pixels Per Frame > 1% Diff.	**Ratio Uniform over Struct.**	**RMS Error**	**Ratio Uniform over Struct.**	**Avg Diff.**	**Max Pixel Diff.**	**Total Samples (nodes)**	**Samples for Struct. Alg.**
Structured	0.6%	-	0.0044	-	1.19	6	3831	2678
Quadratic	5.3%	8.8	0.0081	1.84	1.83	13	3920	-
Linear (w/ Compensation)	6.1%	10.2	0.0093	2.11	1.80	17	6480	-
Quadratic (w/ Compensation)	0.9%	1.5	0.0038	0.86	1.25	7	6480	-

Table 3-1. Test Results for Structured Tensor Product Interpolants (Source Translation)

As can be seen in Tables 3-1 and 3-2, the structured method performs better with respect to all the error metrics used when compared to the uniform quadratic and uniform linear solutions that use about as many samples as used in the tensor product interpolants. Overall the structured algorithm performs better in the case of source translation than in that of source rotation. In terms of pixel difference it performs 8.8 times better than uniform quadratic for the former and 3.9 times better for the latter. In terms of RMS error the improvement is 1.84 and 1.30 times respectively.

	Average% Pixels Per Frame > 1% Diff.	**Ratio Uniform over Struct**	**RMS Error**	**Ratio Uniform over Struct**	**Avg Diff.**	**Max Pixel Diff.**	**Total Samples (nodes)**	**Samples for Struct. Alg.**
Structured	0.7%	-	0.0169	-	1.24	5	3307	3043
								-
Quadratic	2.7%	3.9	0.0221	1.30	1.58	15	3920	-
Linear (w/ Compensation)	2.2%	3.1	0.0233	1.37	1.50	17	6480	-
Quadratic (w/ Compensation)	1.1%	1.6	0.0125	0.73	1.37	7	6480	-

Table 3-2. Test Results for Structured Tensor Product Interpolants (Source Rotation)

When compared to the compensated uniform quadratic solution (Quadratic w/ Compensation), the structured approach still performs better in terms of pixel differences (1.5 and 1.6 times better respectively), but is slightly worse in terms of RMS error (.86 and .73). Note however that the structured method is still better in terms of maximum pixel difference and average pixel difference. It must be underlined that due to its nature as an average, RMS error should be used as one indication in the complete set of test criteria. In addition, visual and consistency issues must also be examined (see next section).

3.7.2.1 Consistency in Animation and Error Distribution

The structured algorithm clearly outperforms uniform grids when additional samples are not taken. Given the restrictions of the current implementation outlined above, the performance, when compensation is made for the additional expense, is less encouraging. Besides the purely numerical results, it is also important to see where and when the error occurs, particularly in the context of animation.

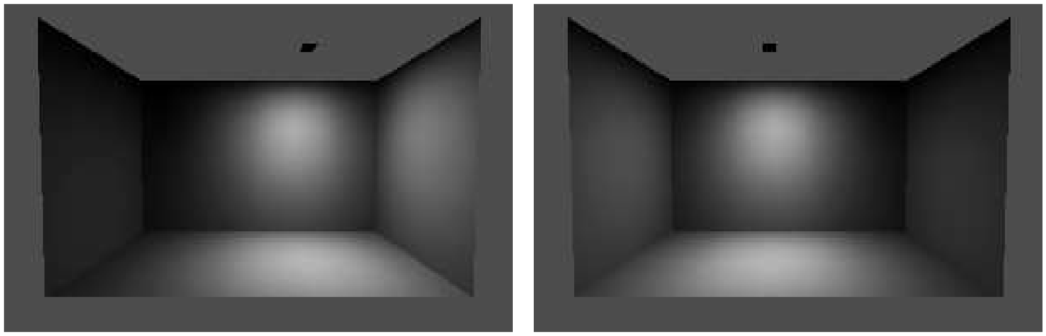


Figure 3-17. Frames 2 & 13 of Test Suite LS Translation (Shaded Structured Solution)

The error due to the structured solution, is *consistent* since the algorithm tracks the real maximum of the radiance function, and modifies the mesh appropriately. In addition, the error tends to occur in the regions of lower power, and is typically a result of a slight over- or under-shooting due to parabolic interpolation in the tail regions. To illustrate this two shaded frames are shown from the structured solution (Figure 3-17) and from the uniform quadratic solution (Figure 3-18). The same frames are shown with the error regions superimposed as dark splotches in Figure 3-19 and Figure 3-20. Figure 3-17 can be found in Appendix C as Colour Images 1(a) and (b). Figure 3-18 can be found as Colour Images 2(a) and (b).

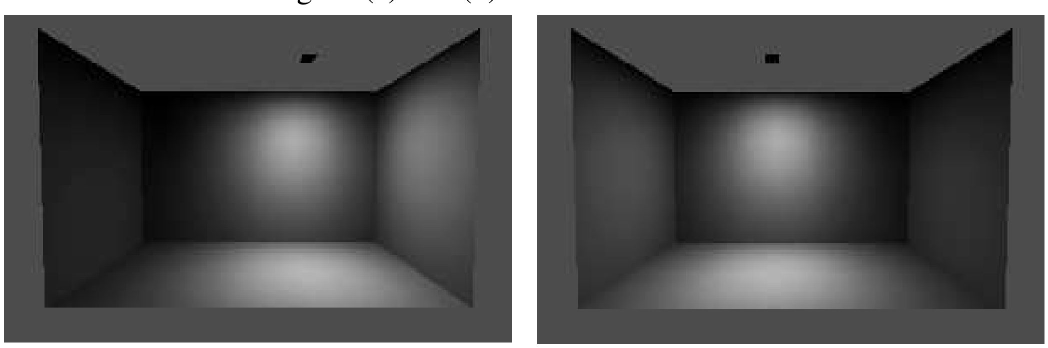


Figure 3-18. Frames 2 & 13 of Test Suite LS Translation (Shaded Unif. Quad. Solution)

Note how for the structured solution the error in the bright regions moves consistently with the motion of the light source. Additional error is mainly in the dark regions (e.g. at top of the back wall on the left of Figure 3-19). Conversely, the uniform quadratic solution may appear to have less error, but the regions of error do not consistently track the maximum, and on the right of Figure 3-20 are particularly evident in the bright regions. Figure 3-19 is shown as Colour Images

4(a) and (b), while Figure 3-20 as Colour Images 4(a) and (b) in Appendix C, where the error is shown in red and can be seen more clearly.

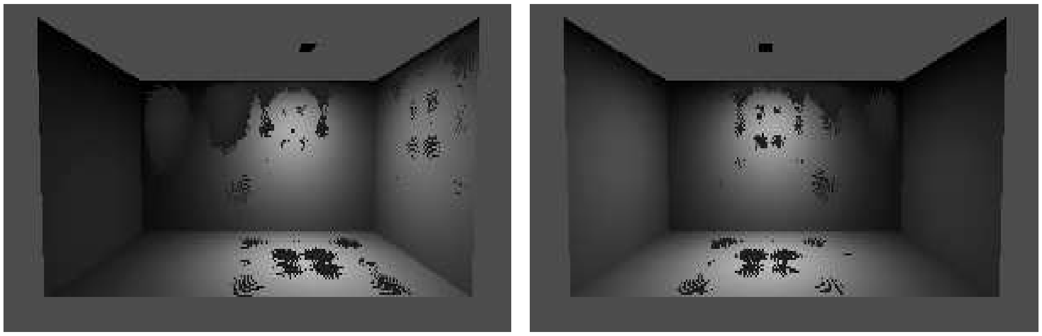


Figure 3-19. Frames 2/13 of Test Suite LS Translation (Struct. Solution w/ Error)

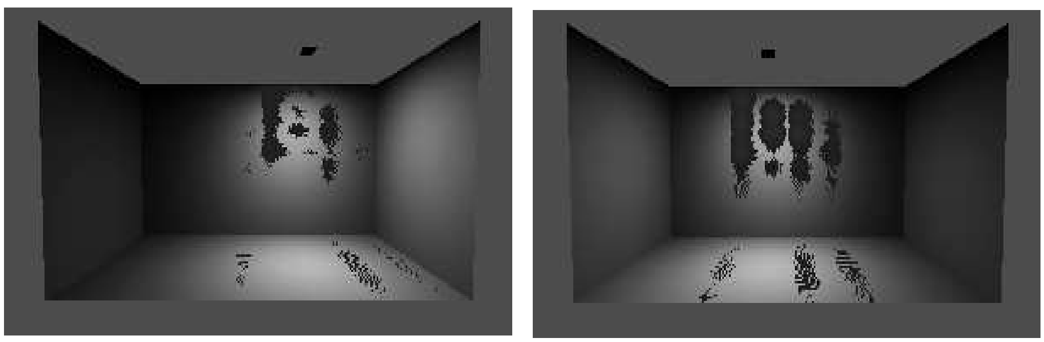


Figure 3-20. Frames 2/13 of Test Suite LS Translation (Unif. Quad. Solution w/ Error)

3.7.3 Test Results for Structured Triangular Grid Interpolants

A preliminary implementation of the triangular grid has been tested. The drawbacks mentioned in the discussion of the approach seem to far overwhelm the possible benefits.

Test results are reported in Table 3-3 for a simple scene containing a single receiver and a single triangular source moving over the surface. For approximately the same number of samples, the algorithm performs much worse by all criteria, for an example case in which the tensor product performs satisfactorily.

Even though further investigation is necessary, it seems that triangular grids require a significant number of additional samples, when compared to tensor product interpolants, to achieve the same quality. In addition the visual quality of the triangular meshes is quite poor, since mach bands appear along the triangular edges where quadratic triangular elements meet.

	Average% Pixels Per Frame > 1% Diff.	**Ratio Uniform over Struct**	**RMS Error**	**Ratio Uniform over Struct**	**Avg Diff.**	**Max Pixel Diff.**	**Total Samples**	**Samples for Struct. Alg.**
Structured (Triangular)	7.90%	-	0.0132	-	1.88	28	5430	3388
								-
Structured (Tensor prod.)	0.04%	0.005	0.0047	0.35	1.10	5	4277	1131
Linear (Compensated)	3.40%	0.430	0.0094	0.71	1.56	13	5120	-
Quadratic (Compensated)	0.20%	0.025	0.0043	0.32	1.25	5	5120	-

Table 3-3. Test Results for Triangular Grid Interpolants

3.8 Discussion and Conclusions

In this chapter, properties of radiance from unoccluded diffuse area light sources have been extensively studied. Radiance functions in these configurations were observed to be smooth and slowly varying in general. The property of unimodality, that is that radiance has a single maximum over a receiver plane and diminishes away from that maximum, was proven for a disc source. It was also conjectured that this property is true for all convex polygonal planar diffuse light sources. The derivatives and curvature of radiance functions were also studied. The curve around the maximum on which the radiance has zero curvature is called the critical curve. The function is convex inside the region of the critical curve and concave elsewhere.

This study of radiance properties led to the development of an algorithm for sample placement and radiance reconstruction. A good approximation to the maximum is found with two simple one-dimensional steps, along two orthogonal axes. The same restriction to two axes is used to find the inflection points along these axes as an approximation to the critical curve. This results in two lists of segments along the two axes in which the radiance function is monotonic and convex or concave. Using these properties, an adaptive subdivision algorithm is used for each of the two lists, segmenting them into linear or quadratic regions that satisfy well defined error bounds.

The two one-parameter lists are then used to build a two-dimensional interpolant. Two formulations are considered, a piecewise polynomial tensor product and a triangular grid. The resulting tensor product grids adapt to movement of the light source. More samples are placed in regions with high variation and power, and fewer samples are placed in smoothly varying, low-power regions. In addition this non-uniform, sparse sampling is guaranteed to satisfy user defined error bounds. A preliminary implementation of the triangular mesh indicates that it does not perform as well as the tensor product approach.

Test results of the implementation of the tensor product algorithm have shown that the algorithm clearly outperforms a uniform grid that contains the same number of sample points as those in the structured grid. When the expense of the structured algorithm is factored in the comparison, given the limitations of the current implementation, the structured algorithm still performs better in 3 of the 4 error metrics used. In addition the structured algorithm is consistent and it tracks important properties of the radiance function, specifically the maximum and the critical curve. As a consequence visual artifacts for animation sequences are much reduced when compared to the uniform grid approach. Finally, the error tends to be located in the low power regions which are less important visually than the regions in the neighbourhood of the maximum.

Chapter 4

Computing Penumbral Boundaries

4.1 Introduction

Interesting computer graphics models include geometries in which the light source is partially or completely occluded, giving rise to subtle shadow regions. In such configurations, the structured sampling algorithm described in the previous chapter is insufficient; the structure of illumination has been further complicated by the introduction of discontinuities due to occlusion. It is thus necessary to characterise the behaviour of illumination in the presence of discontinuities.

Previous researchers have only minimally exploited the structure of illumination functions in the penumbra. Campbell and Fussell [9] were the first to observe that multiple extrema can be found in the penumbral regions. In the discontinuity meshing algorithms to date (Lischinski, Tampieri and Greenberg [39], Heckbert [33]) it is assumed that reconstruction quality will improve when some of the discontinuity lines are identified. In these papers however, detailed characterisation of radiance structure in the penumbra is not attempted. Teller [62] was the first to observe that regions that have the same σ-aspect can be found, but did not present an algorithm to compute a complete mesh containing such regions.

In this chapter and also in Chapter 5, it will become evident that the segmentation of the scene into regions of equivalent σ-aspects is a necessary step for a clearer understanding of the structure of illumination in penumbral regions. The use of the backprojection, which is closely related to the σ-aspect, allows geometric and numerical arguments to be used in the determination of radiance properties in partially occluded regions.

As a consequence of these observations, it is necessary to develop an algorithm that allows the segmentation of a scene into regions of equal σ-aspect. In what follows, only closed, non-interpenetrating polyhedral environments are considered. Such an algorithm has not been completely developed and implemented in previous research. Concurrently with the work of this thesis, the algorithm of Stewart and Ghali [56] was proposed. The advantages and disadvantages of this approach were discussed in Section 2.5.3. Reference to relevant aspects of this work will be made when appropriate.

In Section 4.2, some definitions pertinent to discontinuity meshes and backprojections are presented. In Section 4.3 the new algorithm to compute the mesh is described. Overall, simple algorithms and data structures are used to compute the mesh. Empirical observations of typical scene structure are made throughout the development of the algorithm, that suggest a fast average expected running time, despite pessimistic worst case asymptotic complexity. Statistics are presented that support these observations.

Our algorithm treats all EV-surfaces correctly by improving and completing Heckbert's partial solution [33]. There are two main improvements: first the plane sweep pass of that approach can be avoided by incrementally building the mesh as the edges are inserted; second a voxel-based spatial subdivision structure can be used to significantly limit the number of intersections that need be performed. The algorithm completes the treatment of EV surfaces by identifying and casting the additional EV surfaces that affect the mesh. The spatial subdivision structure is again used to allow this step to be performed efficiently. The new algorithm to treat all EEE discontinuity surfaces is then outlined. First, an efficient method of identifying EEE surfaces that include one edge of the emitter is presented. Second, an algorithm to identify all relevant additional EEE surfaces is developed, as well as the full details of an algorithm for computation and visibility treatment of these surfaces.

In Section 4.4, an efficient incremental algorithm to compute the backprojections is presented. The algorithm assumes that the full mesh has been computed. Subsequently in Section 4.5 results and statistics from a partial implementation of the discontinuity mesh are collected and presented. These statistics substantiate many of the intuitions used to develop the algorithm.

Once the mesh has been computed and the backprojection calculated in each face of the mesh, it is possible to compute accurate images of scenes with area light sources very efficiently. In addition, when the incremental backprojection calculation algorithm is used, the expense of computing the illumination is drastically reduced. In previous approaches this expense was equal or greater than that of computing the mesh. Finally the backprojection is central in the study and comprehension of radiance properties in the penumbra, as discussed in Chapter 5.

4.2 The Backprojection

In Chapter 2, the two main categories of discontinuity surfaces, EV-surfaces and EEE-surfaces were introduced and discussed. The visual events that these surfaces cause were also presented. As a reminder, Figure 2-7 is repeated here as Figure 4-1. These events correspond to discontinuity in the derivatives of radiance as it varies over the surface of a polygon in the scene.

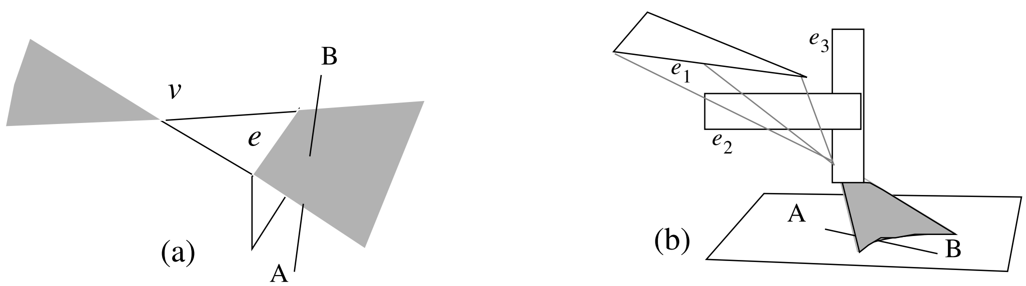


Figure 4-1. EV and EEE Discontinuity Surfaces

In addition, the concepts of an aspect and a σ-aspect were presented, as a way of describing qualitative visibility (Section 2.5.2). The definition of a σ-aspect requires the construction of a labelled drawing and auxiliary graph and is thus unsuited to the requirements of illumination calculations. The more convenient construction of a *backprojection* is presented in the next section.

4.2.1 The Backprojection and the Discontinuity Mesh

The following definition follows that of the σ-aspect graph in Stewart and Ghali [56].

Definition 4-1: *Given a polygonal light emitter* σ *and a scene consisting of polygons* $\{p_i\}$, *the partition of each polygon in the scene into regions that have the same* σ*-aspect, is called the* ***discontinuity mesh*** *of the emitter* σ.

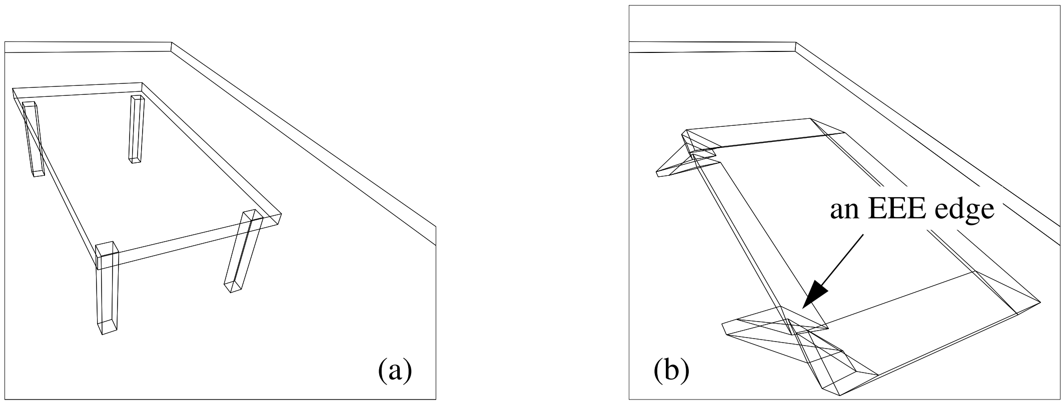


Figure 4-2. A Scene and the Discontinuity Mesh of a Source on the Floor

Each region with the same σ-aspect is called a *face* of the mesh. Previous implemented algorithms compute subsets of this mesh. In particular both Lischinski, Tampieri and Greenberg [39] and Heckbert [33] ignore all EEE events, as well as additional non-emitter EV surfaces that affect the discontinuity mesh. An example of a complete mesh for the floor is shown in Figure 4-2(b), for a scene containing a table and a triangular light source (Figure 4-2(a)). A rendered version from a different viewpoint can be seen in Figure 4-3. The previous algorithms would not have computed the EEE curves due to source edges and the corners formed from the table legs and side, shown in Figure 4-2(b).

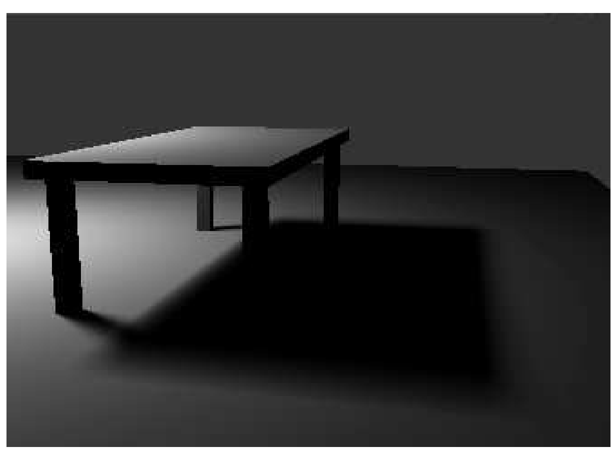


Figure 4-3. Table Scene

The visible area of the source from a point P forms a set of polygons, whose vertices are either formed by the projection of environment edges onto the source or are vertices of the original source polygon. Assume that we have planar polygonal emitter σ in a scene, and the discontinuity mesh of that emitter. For each face in the discontinuity mesh, we define the backprojection instance and the backprojection as follows.

Definition 4-2: *A* ***backprojection instance*** *at a point P within a face of the discontinuity mesh, is the set of maximal polygons on* σ *that form the visible region of* σ *from P.*

The vertices of these polygons are either emitter vertices or intersections of projections of edge pairs onto the plane of σ through the point P. We call the emitter vertices or edge pairs *elements*. We can thus represent the σ-aspect in a face as a set of ordered lists of such elements, which we call a *backprojection*.

Definition 4-3: *The* ***backprojection*** *of a face is the set of ordered lists of emitter vertices and edge pairs, such that at every point P in the face, the projection through P of these elements onto the plane of* σ *form the backprojection instance at P.*

A related but different definition of a backprojection was given by Stewart and Ghali in [56]. The lists of a backprojection contain three element types, which we call “bp-elements”. Examples of each bp-element type, and the resulting vertices for a backprojection instance, are shown in Figure 4-4. The three types are listed below.

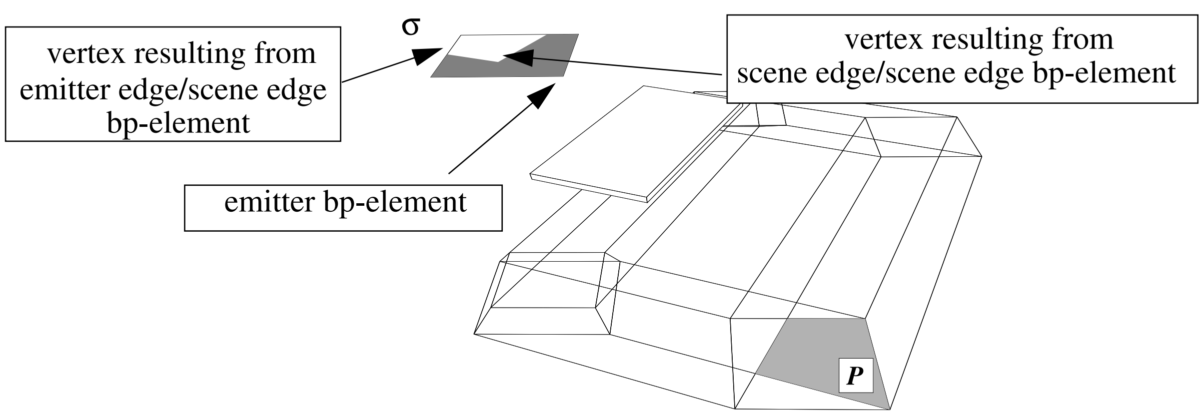


Figure 4-4. The Bp-Element Types

1. Emitter bp-elements, which are vertices of the emitter.
2. Emitter edge/scene edge bp-elements, consisting of an emitter edge and a scene (non-emitter) edge. The projection of a scene edge through a point P in a face intersects the emitter edge to form a polygon vertex of the backprojection instance.
3. Scene edge/scene edge elements consisting of two scene (non-emitter) edges. At a point P in a face, the projection through P of the intersection of the two edges onto the plane of the emitter forms an interior vertex of the polygons of the backprojection instance.

Type 1 elements are called invariant elements, while Type 2 and 3 are called variable elements.

A Notation for Backprojections

Since it is necessary to describe backprojections in the incremental algorithm presented in Section 4.4.2, the following notation is introduced. Type 1 bp-elements are denoted by the name of the emitter vertex in parentheses, e.g., (v). Type 2 bp-elements are denoted by the ordered sequence (emitter edge, scene polygon edge). Type 3 bp-elements are denoted by the ordered sequence (scene edge, scene edge). The following convention is used to distinguish between Type 2 and Type 3 bp-elements: emitter edges start with the letter e and scene edges start with the letter b. Thus (e_1, b_1) is an emitter edge/scene edge bp-element, while (b_1, b_2) is an scene edge/scene edge bp-element. Finally a sequence of the form: (..) ; (..) ; (..), denotes part of the lists of bp-elements.

Computing the Backprojection Instance

To compute the instance of a backprojection at a point P, each edge of a variable bp-element is projected through the point P onto the emitter plane, to a single, fixed three-dimensional point.

To perform this projection at a given point P, for an emitter edge/scene edge pair, the plane formed by the scene edge and point P is intersected with the emitter edge. The resulting point of intersection is the fixed vertex. Similarly, to find the vertex due to a scene edge/scene edge pair, the two planes formed by the point P and each of the two edges are formed. The intersection of these two planes forms a ray. The intersection of the ray and the emitter plane is the desired instance vertex.

Thus, given the representation of a backprojection by the relevant edge pairs, and the specific point *P*, the list of fixed vertices that form the visible portions of the light source can be formed relatively cheaply. Eq. (2-26) can then be applied for each fixed polygon to calculate the value of illumination at a partially occluded point *P*.

Backprojections and the Discontinuity Mesh

Backprojections are a useful tool for reasoning about illumination in penumbral regions. As shall be seen in the following chapter, the efficient determination of the radiance function within mesh faces using the backprojection facilitates the development of structured sampling approaches for penumbra.

4.3 An Efficient Algorithm for Computing a Complete Mesh

As mentioned above, both previously implemented discontinuity meshing algorithms compute only a partial mesh. As a result, the σ-aspect is not always the same at all points within a single face of the meshes generated using those algorithms. In this section a complete algorithm is presented in detail.

In the following, a *feature* is either an edge or a vertex. Also, any EV surface is said to form a (planar) *wedge*. A *shadow edge* of a polyhedron with respect to a point is an edge that is contained in the silhouette of that polyhedron when viewed from that point. A *shadow vertex*, is a vertex which is attached to at least one shadow edge.

Before computing and casting discontinuity surfaces into the environment, it is first convenient to compute the lines at which objects touch[1]. These have been called D^0 events by Heckbert [34]. To compute these efficiently, each object in the environment is visited, and its neighbours are found using a voxel-based spatial subdivision structure (see Section 4.3.2.1). Lines are inserted on the faces of objects that touch, denoting the discontinuities of value that any illumination function will have at these boundaries of contact.

1. Recall that the environments under consideration do not include interpenetrating polyhedra.

The discontinuity surfaces that need be considered to calculate the complete discontinuity mesh for one emitter are of the following four types: (a) EV surfaces in which one feature is on the emitter, called emitter-EV surfaces, (b) EV surfaces that do not include an emitter feature, but whose plane intersects the emitter (non-emitter EV surfaces), (c) EEE surfaces that contain an emitter edge, which we call E_eEE surfaces, and (d) other EEE surfaces that do not contain an emitter edge, called non-emitter EEE surfaces.

By separating the events into categories (a)-(d), it is possible to compute the complete mesh efficiently and without significantly more complicated algorithms. The overall strategy is to treat simple events with simple and efficient algorithms, and leave the more complex and infrequent cases to separate, more involved approaches.

The algorithms chosen to perform the subtasks use simple data structures and solution methods. As a consequence the worst case asymptotic complexity of the algorithm components can be extremely pessimistic. Many computer graphics environments however tend to have such properties that these worst cases do not occur often, if at all. At each point in the description of the algorithm, the assumption about scene behaviour will be made, and the effect on the resulting complexity presented. Statistics run on typical scenes will be presented to support the assumptions. Detailed presentation of these statistics is given in Section 4.5.

Before discussing the treatment of each category of discontinuity surfaces, the extended topological data structure introduced to build the discontinuity mesh incrementally is discussed in Section 4.3.1. In Section 4.3.2 the improved and complete algorithm to treat emitter and non-emitter EV-surfaces is presented. In Section 4.3.3 the new algorithm to treat EEE surfaces is detailed.

4.3.1 An Extended Topological Data Structure

A topological data structure is used to store the mesh. This structure is similar to the winged-edge structures described by Glassner [23]. The structure stores vertices that are connected by edges. These edges enclose faces that can contain edge cycles, possibly nested. This face-edge-vertex structure maintains consistent adjacency information. Each edge is stored with a twin edge in the opposite direction, and a left and right face pointer. The meaning of left and right faces is reversed for the twin edge.

During the computation of the penumbral boundaries, edges can be added to the mesh in such a way that the faces of the mesh are not closed. To allow for this incremental construction, the standard face-edge-vertex structure has been augmented to handle the intermediate configurations that arise as the mesh is being built.

Each face of the mesh is bounded by an edge cycle, called a *face boundary*. A *chain* of edges is a sequence of edges that do not necessarily form a closed loop. In addition, the following special types of edges are defined:

- Lonely edges, that are not connected to a face boundary.
- Dangly edges, that are connected to a face boundary but are not in any simple cycle.

The left and right faces of a lonely or a dangly edge are the same, thus maintaining consistency.

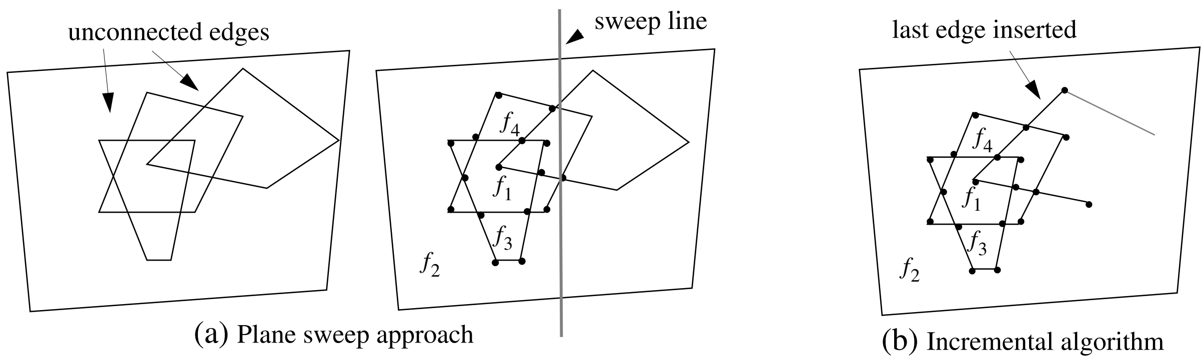


Figure 4-5. Previous and New Approaches to Mesh Construction

Computing the Arrangement of Edges on a Receiver

In Heckbert's approach [33], the edges resulting from the intersection of the EV-surfaces with the receiver objects are associated in an unconnected fashion with the receiver surfaces. A plane sweep step is then required to connect these segments, shown in Figure 4-5(a). As the sweep line passes the points, intersections and face structures are built. This step can be expensive and can prove unrobust in practice. As an alternative to this approach, in our algorithm the mesh is built incrementally as the propagation of discontinuity surfaces progresses.

EV surfaces are formed by the edges of a *blocker* polygon and a vertex either of the emitter or a different polygon (see for example Figure 4-16). These edges are traversed in order of the blocker polygon boundary. After each edge insertion, the receiver surface stores the edge as the "last edge inserted". When the next insertion due to the same blocker occurs, it will be connected to the previous vertex (an endpoint of the previous edge), and thus no search time is required to locate the face in which the new edge will be inserted (see lines 2 and 4 in the pseudo-code of Figure 4-6). In Figure 4-5(b) for example, the dotted line is the newly inserted edge. The previous endpoint is the new starting point for the line, and no search is required to find the face in which the new edge is inserted.

When a new blocker is processed, the line connecting the previous insertion point to the current insertion point is followed. In the early stages, no lines will be crossed, and thus the face of insertion will be found without a search. In the later stages, it is possible that a somewhat larger portion of the mesh structure will need to be searched. On average, it is expected that only a small number of faces will be searched in this fashion. This search is performed by the procedure `find_face()` in line 3 of Figure 4-6.

During this incremental construction, configurations in which the mesh contains lonely and dangly edges often occur. The augmented edges are necessary to maintain a consistent concept of faces at any moment. Care must be taken when a chain is being "closed" to form a closed face, so that all dangly and lonely edges are associated with the correct face.

It must also be noted that in general the insertion of an edge into the mesh involves the intersection of the corresponding line segment with the edges of the faces that lie between the two endpoints. Similarly to above, it is expected that only a small number of faces are traversed in such insertion operations. This face traversal stage is performed by the function `insert_edge_in_face()` in Figure 4-6.

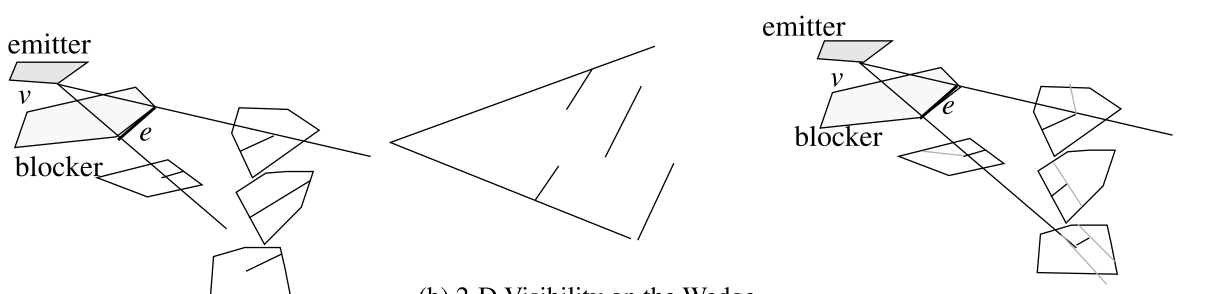
```
insert_edge( Edge2d e, Polygon receiver, Polygon blocker )
{
    Edge2d last_edge = receiver->last_edge_inserted();
    Face f;

    if( receiver->get_last_blocker() == blocker &&
        last_edge->head_vertex() == e->tail_vertex() )           1.
        f = last_edge->left_face();                              2.
    }
    else
        f = find_face( last_edge->head_vert(), e->tail_vert() ); 3.
    insert_edge_in_face( f, e );                                 4.
}

insert_edge_in_face( Face f, Edge2d e )
{
    Face face_to_insert = f;
    done = FALSE;
    while ( !done ) {
        if( e intersects an edge of f at inter_pt ) {            5.
              Edge2d enew;
              enew.set( e->tail_vert(), inter_pt );              6.
              insert( f, enew );                                 7.
              f = next_face_after ( inter_pt );                  8.
              e.set( inter_pt, e->head_vert() );                 9.
        }
        else {
              insert(f, e);                                      10.
              done = TRUE;
        }
    }
}
```

Figure 4-6. The New Edge Insertion Algorithm

The plane sweep approach in Heckbert's algorithm has an $O(s \log s)$ expense of connecting the edges, where s is the size of the mesh (i.e., the number of faces in the mesh). This expense is always incurred in that algorithm. In the worst case, the new algorithm will require $O(s^2)$ operations, since for every insertion (there are s such insertions, assuming a bounded number of edges per face), the whole mesh could be traversed. In practice, the number of faces traversed to locate the face of insertion is expected to be significantly smaller than *s,* as is the number of faces traversed for intersections during edge insertions. Statistics gathered in Table 4-1 of the number of faces intersected when inserting an edge support this claim. For scenes where the size of the mesh was 203, 1372 and 2944 faces, the average number of faces crossed (in `insert_edge_in_face()`) while inserting an edge was 1.8, 1.4 and 1.42 respectively. Thus the expense of the arrangement of edges is expected to be lower than that of the plane sweep, with the additional benefit of more robust insertions in practice. The incremental algorithm is more robust than the plane sweep since, by maintaining a pointer to the previously inserted edge, the connections during edge insertion that form mesh faces are less sensitive to numerical problems.

Augmenting the Mesh to Include Curved Edges

Since EEE surfaces are being treated, it is necessary to store curved edges as boundaries of faces. This can be achieved by augmenting the edge data structure to include a “curved” flag. The equation of the quadratic curve (see Eq. (4-1) in Section 4.3.3.2) is stored for such edges and any operation to determine point in face inclusion or line/edge intersections uses this equation instead of that of a line. When inserting a curved edge, care must be taken to traverse the correct sequence of faces, splitting the relevant edges at the points where the curve intersects the (possibly curved) edges of the mesh.



(a) Initial Casting (b) 2-D Visibility on the Wedge (c) Final Insertion in Meshes

Figure 4-7. EV Surface Casting

4.3.2 Computing the EV Surfaces

Our algorithm to compute all EV surfaces extends and improves Heckbert’s partial solution [33]. The first set of improvements involve the augmented data structure described above. The second is the use of a voxel-based spatial subdivision structure to reduce the number of intersections of the EV surfaces with the objects in the environment significantly. The algorithm is completed with the extension to non-emitter EV surfaces. The spatial subdivision structure is used to accelerate the computation of these additional events as well.

4.3.2.1 Casting Emitter-EV Surfaces Through the Environment

To generate the edges of the discontinuity mesh due to emitter EV surfaces, the general structure of Heckbert's algorithm [33] is followed.

For each vertex of the emitter and every other shadow edge in the environment (e.g. v and e in Figure 4-7(a)), and for each edge of the emitter and every shadow vertex in the environment a wedge is formed. The intersection of each wedge with every polygon in the environment is computed. For each such wedge, the line segments corresponding to these intersections are then transformed onto the wedge plane, and inserted into a sorted list (Figure 4-7(b)). A two-dimensional visibility step is performed that results in the correct segments being inserted into the meshes of the surfaces intersected. These are represented as thick lines in Figure 4-7(c). The visibility algorithm is a modified two-dimensional Weiler-Atherton algorithm [4].

Special care is taken to accommodate lines that are colinear within machine precision. Specifically, coplanar or almost coplanar wedges are identified beforehand and the mesh edges caused by such wedges are collapsed into one. Additional colinear edges are identified at insertion time. This could potentially cause problems of robustness. In the general case, a consistent algorithm is required to guarantee the construction of a well-formed mesh.

Maximal edges are used to determine the boundary between light and penumbra, as described by Campbell and Fussell [9]. For example, for a specific blocker edge, and a source with p vertices, p wedges are formed. At least one of these wedge planes will have all the other wedges in its negative half-space, and is thus maximal. Only edges inserted in the mesh due to wedges of this type may be part of the boundary between light and penumbra. After computing the mesh, the faces are searched to determine those faces whose edges are all maximal. These faces are completely in light, and are well defined.

In a manner similar to that for the maximal edges, minimal edges can be found, which will be the portions of the boundary between umbra and penumbra that are lines. However, some parts of the umbra/penumbra boundary may be EEE curves.

A special case occurs if no curves are inserted into a mesh of a polygon. When each such polygon is processed as a blocker, if nothing has been inserted into its mesh, it is determined if the wedges from the emitter are blocked. If they are not blocked, the polygon is in light, else it is in penumbra or umbra.

Complexity and Acceleration Techniques for Emitter-EV Computations

Since we need to compute the intersection of each EV surface with every object in the environment, the cost of casting one wedge is $O(n)$, where n is the number of objects. Let m be the size of the set of resulting intersections. In most computer graphics environments, the number m of objects a single wedge will intersect is typically much smaller than n, since the wedge is a planar polygon limited by the size of the source. The cost of the visibility step is $O(m^2)$. Therefore the total cost of the emitter-EV surface processing (ignoring insertion) is $O(n(n+m^2))$ which in practice is close to $O(n^2)$, since m is insignificant compared to n. These assumptions are supported by statistics shown in Table 4-1 in which, for test scenes with n equal to 55, 187, 331, 409, and 661, the values of m are 1.51, 3.42, 2.86, 3.27 and 2.88 respectively. The running time to compute the mesh (*Time (NoGrid)*) for a scene of 331 polygons is 538.2 seconds, while for 661 polygons, which is double the size, the running time more than quadruples to 2395.9 seconds.

In the ray tracing approach developed for linear light sources by Poulin and Amanatides [46], a regular grid (such as that developed by Amanatides and Woo [2]) is used to optimise the intersection of objects with the triangle formed by the visible point and the linear source (see also Franklin [21]). The voxel grid is a regular subdivision of space into cubes, called *voxels*. The environment is preprocessed once, at which point the objects are inserted into the voxels which contain them or which they intersect. Thus when traversing a subset of the scene, the objects in that subset can be found just by examining the voxels traversed, instead of searching the entire object list.

The optimisation for the linear light source can be adapted easily for the intersection of an EV wedge with the objects in the environment. After the wedge is formed from the vertex and the corresponding edge, the wedge is extended to infinity, and clipped to the limits of the bounding box of the environment. The resulting polygon is then scan converted on the voxel grid, and a set of candidate objects is created. By reducing the candidate set to a much smaller expected number, the casting time for all emitter-EV wedges may approach $O(n)$ in practice, instead of $O(n^2)$ for the class of scenes studied. This is supported by the results (see Section 4.5), where a scene containing almost twice as many polygons as a previous scene takes twice as long to compute. It must be added that the voxel spatial subdivision is limited by the fact that for very densely populated scenes, high subdivision may be required, resulting in higher memory requirements and an added cost of traversing the voxel structure. Similar issues are discussed by Poulin and Amanatides [46].

4.3.2.2 Non-emitter EV Surfaces

Of all possible EV-surfaces, only those that interact with (i.e., cut) the emitter polygon need be considered. An efficient algorithm to identify these surfaces is presented next.

At each vertex v in the environment, a pyramid is formed that has v as its apex and the emitter polygon as its base (see Figure 4-8(a)). The pyramid is also extended to the other side of the vertex. Any objects that intersect this pyramid are inserted in a list. The list is then traversed, and every edge is tested to see if the EV wedge formed with v cuts the source. These additional EV surfaces are collected into a list of candidate non-emitter EV wedges.

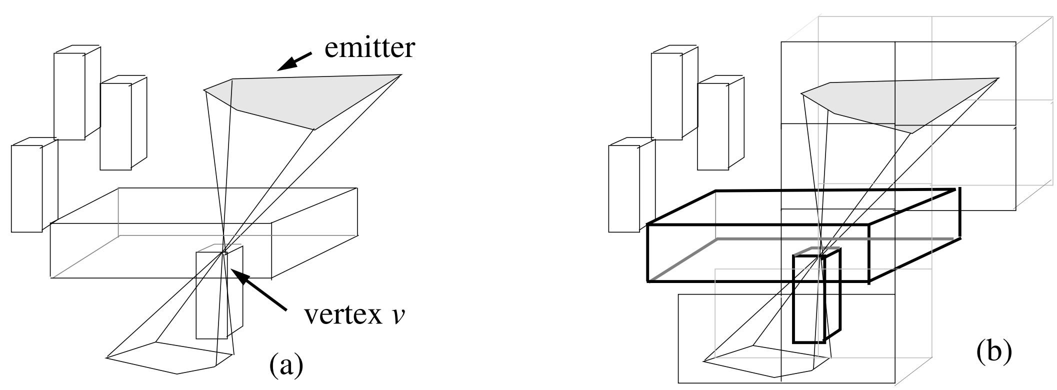


Figure 4-8. The EV Pyramid and the Grid

After the candidate list is formed, the non-emitter EV surfaces in the list are cast, using a modified version of the algorithm used to cast the emitter EV surfaces. The modifications required are as follows. First the wedge is clipped to the limits of the source before casting begins. Second, it is necessary to modify the algorithm so that edges are inserted into the mesh only in the region of the wedge that is past the edge or vertex, whichever is furthest from the emitter. This is not necessary for emitter-EV surfaces, since the corresponding region in which the edges should be inserted is delimited by the feature that does not belong to the emitter (edge or vertex respectively).

Acceleration Using the Voxel Structure

The creation of the candidate lists of non-emitter EV events for all vertices, if performed naively, would cost $O(n^2)$ time, where n is the number of edges in the environment. This is true because at each vertex, all the edges would have to be checked to determine whether they are within the pyramidal volume described above and shown in Figure 4-8(a).

Using the spatial subdivision structure described previously, the voxels which are either internal to the volume or are cut by the bounding planes are found. Subsequently, only the wedges formed by v and the edges of these objects are tested to determine if they cut the source. For each vertex, the number k of edges examined is on the average expected to be much smaller than n, where n is the total number of edges in the scene. This is supported by statistics run on typical scenes, shown in Table 4-2. For a scene of 372, 696 and 1272 edges, k was 25.59, 29.88 and 36.75. In Figure 4-8(b) for example, only the thick edges are actually examined. All other edges are not in the voxels contained or cut by the volume, and are thus not examined.

In addition, the total number of non-emitter EV surfaces can be small compared to the number of emitter-EV surfaces, depending on the source size. For a scene with a small source, for a total of 2598 emitter-EV surfaces, there were only 232 non-emitter EV surfaces, while for the same scene with a larger source, placed purposely to increase the number of such events, for a total of 3462 emitter wedges, the number rose significantly to 1704 (see also Table 4-2).

4.3.3 EEE Surface Computation

EEE surfaces have not been treated in the two previous implemented discontinuity meshing algorithms ([33][38]). It was stated in those papers that the expense of computing EEE events overwhelms the benefit of their computation. Although this may be true for certain cases, in many cases these events can be important. In addition, their computation is required for the incremental backprojection algorithm, and of course for the computation of the complete mesh. Teller [62] describes the computation of extremal EEE surfaces for a different application, but does not present a complete visibility processing algorithm for these surfaces.

In this section, the treatment of EEE surfaces is described following the separation into two categories described previously; EEE surfaces including an emitter edge, called E_eEE surfaces, and those surfaces not including an emitter edge, that nonetheless cut the emitter polygon, called non-emitter EEE surfaces.

4.3.3.1 E_eEE Surfaces

The most common EEE curves are caused by discontinuity surfaces that include an emitter edge. These events occur whenever there are more than two skew edges in a scene. It is therefore important to identify them efficiently. The following algorithm will work for a large number of cases, without incurring significant additional computational expense. Failure is identified, and the corresponding surfaces are processed correctly using the more involved algorithm presented in the next section.

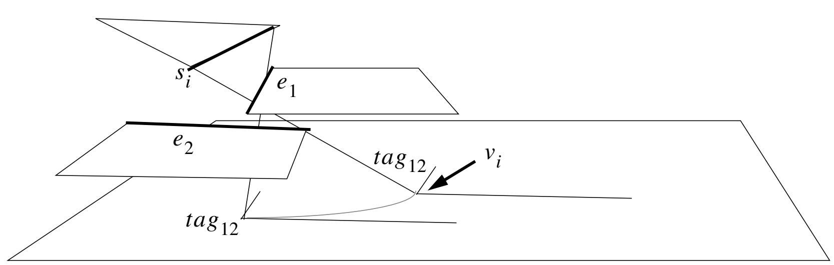


Figure 4-9. Tagging for E_eEE Curves

During the processing of EV surfaces, each vertex receives an integer tag, indicating its generator edges (see Figure 4-9). When two non-adjacent, skew edges meet at a vertex in the mesh, the resulting mesh vertex v_i is inserted in a list. Each v_i was generated by an emitter vertex s_i, so the

EEE conics defined by the triples $\{e_i^s, e_1, e_2\}$ and $\{e_{i+1}^s, e_1, e_2\}$ are drawn onto the surface. These quadratic curves either join vertices with the same tag, or are appropriately clipped. This approach can fail if the mesh vertex v_i is occluded, but portions of the curve are visible. This case can be identified when a curve is clipped to an occluding edge of a polygon or a mesh edge. The more general solution for non-emitter EEE surfaces can then be applied. Situations such as these may occur when small objects are appropriately placed behind corners formed by skew edges. Such occurrences are rare in scenes in which the source edges are small, since the valid portion (see next section) of the discontinuity surface is small.

4.3.3.2 Non-Emitter EEE Surfaces that Cut the Emitter Polygon

To compute a complete mesh, it is necessary to identify and process EEE surfaces formed by three edges that do not include an edge of the emitter, but that cut the emitter polygon. For simple scenes, and scenes of moderate geometric complexity in which the emitter polygons are small, such events do not occur at all. This is true because not only does the quadric surface defined as a ruled surface need to cut the emitter polygon, but the portion that cuts the emitter needs to be composed of lines that touch the interior of all three defining edges. Although this is difficult to visualise, it will become evident that this does not occur very often for the aforementioned scenes.

In the rest of this section, the preliminary mathematics for the quadric surfaces and their parameterisations are presented. Similarly, the treatment of the quadratic curves that result from the intersection of the surfaces with the polygons is discussed. Once these preliminaries are in place, an algorithm to identify the non-emitter EEE surfaces is developed. Finally the details of EEE surfaces casting and visibility processing are presented, as well as the use of the voxel spatial subdivision structure to accelerate both the identification and the casting of the surfaces.

The Quadric Surface, its Parameterisation and the Determination of the Valid Region

The equation of a general quadric surface is given as follows:

$$Ax^2 + By^2 + Cz^2 + Dyz + Exz + Fxy + Gx + Hy + Iz + J = 0 . \tag{4-1}$$

The coefficients of the quadric formed from the three edges $\{e, e_i, e_j\}$, can be determined from the equations found in Gigus, Canny and Seidel [26] or Salmon [50]. We present the derivation of these coefficients in Appendix B.1.

For every point P known to be on the ruled surface defined by edges $\{e_1, e_2, e_3\}$, there is a unique line contained on that surface that passes through edge e_1. If e_1 has points a_1 and b_1 as its endpoints, then any point P_t on the line can be written:

$$P_t = a_1 + t(b_1 - a_1), \qquad t \in [0 \ldots 1] \tag{4-2}$$

The value of the parameter t can thus be used to characterise the point P on the surface given the other two edges. In the following, the parameter t is called the first edge parameter. To compute the value of this parameter for a point P known to be on the surface (e.g. a point on one of the edges), the following operation must be performed. First, form the two planes containing P, e_2 and P and e_3[2]. Next, compute the intersection of the ray defined by the intersection P_i of these two planes with the line equation of e_1. The value of t can then be found by solving Eq. (4-2) for $P_i = P_t$. This construction is shown in Figure 4-10.

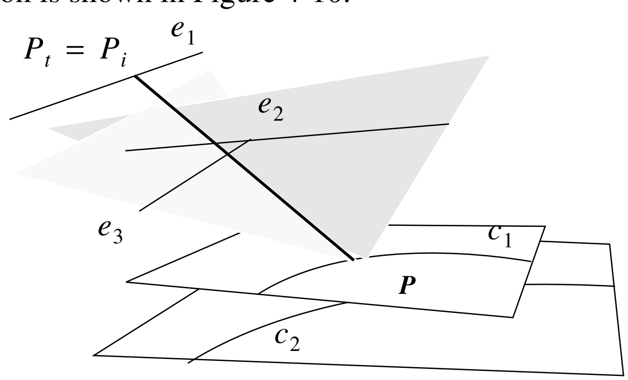


Figure 4-10. Finding the First Edge Parameter

The only portion of the quadric that is valid is that for which the parameter t, corresponding to the ruled lines of that portion, is between 0 and 1. In a similar manner, the regions of ruled lines must also lie within the limits of the two other edges e_2 and e_3.

To determine the valid region, the first edge parameter is found for the endpoints a_2, b_2 of edge e_2 and for the endpoints a_3, b_3 of edge e_3. Call these parameter values t_{a2}, t_{b2}, t_{a3}, t_{b3}. The minimum

2. If P is on e_2 or e_3, one plane is formed with the other edge (e_3 and e_2 respectively) that is then intersected with e_1 to find the parameter value.

common intersection of the intervals $[0, 1]$, $[t_{a2}, t_{b2}]$, $[t_{a3}, t_{b3}]$ is found. If this interval is empty, there is no valid region of the EEE surface and no further processing is required. If such an interval exists, it defines the portion of the surface which is valid. It is possible that the valid region will be disjoint (Teller [63]).

Processing the Quadratic Curves on the Receiver Polygons

Gigus and Malik [25] mention that the quadric of Eq. (4-1) is in general a hyperboloid of one sheet, or a hyperbolic paraboloid if the three edges are parallel to a plane. Thus, the intersection of a plane with the surface can either be a hyperbola, an ellipse or a parabola. To compute the curve of intersection of the quadric with the plane of a polygon, the three generating edges are first transformed into the coordinate system such that the polygon is embedded in the $z = 0$ plane, without loss of generality. The quadratic curve is now immediately given from Eq. (4-1), by setting $z = 0$:

$$Ax^2 + By^2 + Fxy + Gx + Hy + K = 0, \tag{4-3}$$

using the same equations to generate the coefficients. Using a standard set of transformations (Rogers and Adams [47]), the quadratic is transformed into standard form. Suitable parameterisations exist for the ellipse, the hyperbola or the parabola in the standard form (see Rogers and Adams [47], pp. 219-230).

The quadratic curve described by Eq. (4-1) can now be intersected with the edges of the receiver polygon, in the $z = 0$ plane. The result is a collection of segments, possibly clipped to the edges of the receiver polygon. At each endpoint p_i of each segment, two parameter values are computed: the value of the parameter of the curve s_i computed using the parameterisations mentioned above, and the value of the first edge parameter t_i (see Figure 4-11(a)).

To determine the valid portions of the curve segments, the segments are sorted by the curve parameter *s*. In Figure 4-11(a), the segments are (s_1, s_2), (s_3, s_4), (s_5, s_6), and the valid region for the first edge parameter *t* is [0,1]. Segments for which the first edge parameter t_i is not valid for both endpoints (in the same direction e.g. both > 1 or both < 0) are immediately rejected (e.g. (s_1, s_2)). Segments which are completely within the valid region (e.g. (s_5, s_6)) are accepted. For the others, the segment is clipped to the closest endpoint of the valid region (e.g. for (s_3, s_4) the

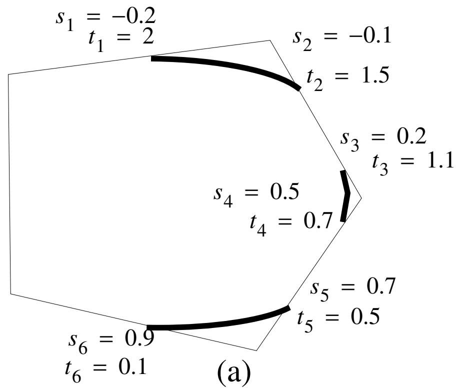


Figure 4-11. Quadratic Curve Processing

segment is clipped to the point for which t_3 is equal to 1). To find the clipped point, the ruled line is generated in the manner similar to that used to find the parameter value, and its intersection with the receiver polygon found. The final clipped segments are those associated with the surface. These are the remaining thick lines shown in Figure 4-11(b).

Identifying the Non-Emitter EEE Surfaces

For each edge e, a volume is formed between the edge and the convex hull of the emitter polygon (see Figure 4-14(a)). This polyhedral volume is used to determine which objects are between the edge and the source, and thus potentially can participate in non-emitter EEE surfaces.

First all shadow edges that are not clearly outside the volume are inserted into a list. All edges that are parallel to e are rejected. There are now k edges $(e_1, \ldots, e_k)$ left in the list. For each non-parallel pair of edges e_i, e_j, a first trivial test is performed: if all the vertices of the emitter polygon have the same sign when their coordinates are substituted into Eq. (4-1), the EEE surface is rejected, since it cannot cut the emitter polygon. A large number of surfaces are rejected in this fashion.

If some vertices have opposite signs, the quadratic curve formed by the intersection of the quadric with the emitter plane is formed. If the region of the quadric cutting the emitter polygon is valid (determined using the method described above), the triple of edges is inserted into the candidate non-emitter EEE surface list.

Given the above discussion, all the machinery is in place to allow the processing of the non-emitter EEE surfaces. Once the candidate list has been filled, each EEE surface in the list is cast. The casting algorithm is equivalent to the algorithm used to cast EV surfaces described in Section 4.3.2.

The intersection of the EEE surface with the polygons in the scene is computed as described above. The valid segments are stored in a structure including their endpoints, the object they are associated with and the parameter values s and t for each endpoint. Given this information, at each endpoint p_i the distance d_i along the ruled line from the first edge can be calculated for each endpoint. The space defined by the coordinates (t_i, d_i) can be used to perform visibility processing.

Each curve segment now corresponds to a line in parametric (t, d) space (see Figure 4-12(a) and (b), where the arrows show the direction d increases). Note that the lines do not correspond to the exact distance at every point in the curve, only at the endpoints of the segments. Relative ordering is maintained because the environment does not include interpenetrating polyhedra.

Once the transformation has been performed, a line sweep visibility processing algorithm can be run on the segments calculated. The segments are first sorted by increasing first edge parameter t.

Each segment has a structure containing "new segments" as they are inserted during the line sweep. A line parallel to the d axis is swept across the plane, and a list of "active" segments is maintained. A segment is active when its starting point has been encountered but not its end. Whenever an endpoint is encountered, an event occurs. At a point just past the endpoint, in terms of increasing t, the ruled line is generated and its intersection with the active segments is computed. The examples referred to below are shown in Figure 4-12(b).

If the event is a starting point, two actions can be taken.

1. On the segment just entered, if the point is closest along the ruled line, a new segment is inserted. This occurs for example for curve c_1 at event v_2.
2. On any other segment, if the point is not the closest, the current new segment is closed. This occurs for curve c_2 at event v_2.

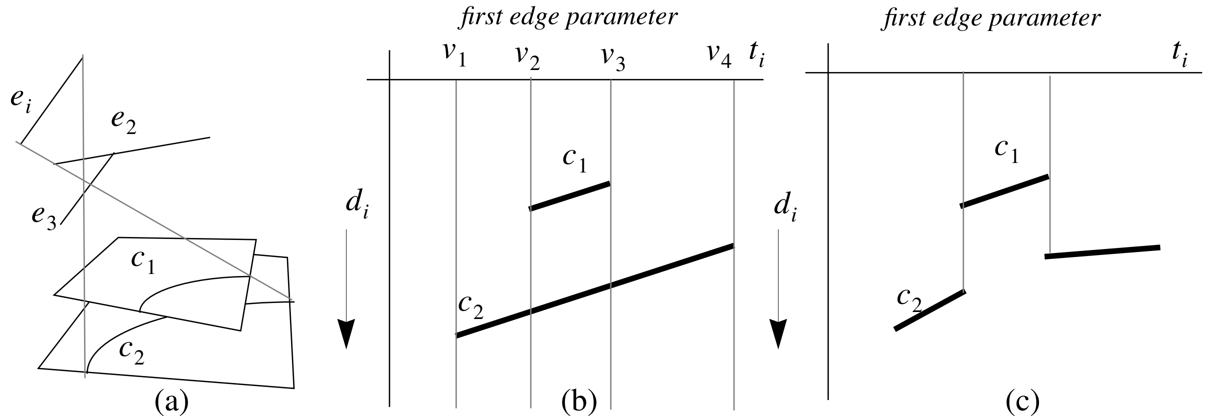


Figure 4-12. Processing the EEE Surfaces

If the event is an endpoint the following action can be taken.

1. On the segment just exited, if there is an active new segment, it is closed. This occurs for curve c_1 at event v_3.
2. On any other segment, if there is no active new segment, a new segment is started. This occurs for curve c_2 at event v_3.

When a segment is opened or closed, the new intersection of the ruled line and the polygon is calculated, corresponding to the first edge parameter of the event. This explains the fact that the new segments shown in Figure 4-12(c) do not lie on the same "line" as the original representation of the curve in (t, d) space. We show next how ordering is guaranteed to be correct since the environment consists of closed non-interpenetrating polyhedra.

Lemma 4-1: *Given two curved segments resulting from the intersection of two EEE quadric surfaces on two different polygons in an environment of non-interpenetrating polyhedra, relative ordering of the two segments is maintained during visible curve processing, performed in (t, d)-space.*

Proof Sketch. We sketch the proof of this lemma referring to Figures 4-12 and 4-13. If the lemma were false, relative ordering of two lines in (t, d) space would change after a visibility step. This is illustrated in Figure 4-13(a)-(b), where from an original configuration in which c_1 was closer (in terms of distance d_i) than c_2 before visibility processing. After a visibility step, an endpoint of

c_2 has a smaller distance d_i than c_1, along at least one ruled line of the quadric surface. If this were to occur, then the polygons, say p_1 and p_2, corresponding to curves c_1 and c_2, would be interpenetrating (see Figure 4-13(c)), since there would be at least one point at which p_2 would cut through p_1.□

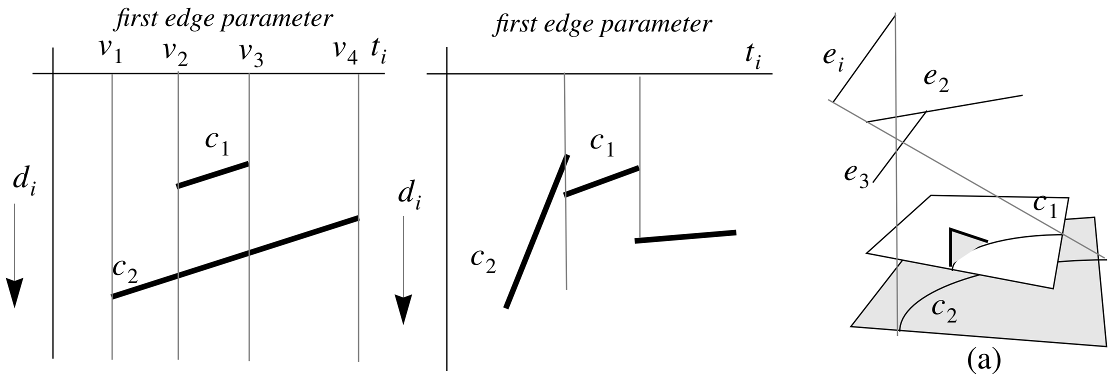


Figure 4-13. Relative Ordering is Maintained in (t, d)-Space

After the visibility step, the curve segments associated with objects that are past the third edge in the sense of the distance *d* are inserted into the corresponding meshes.

Accelerating EEE Surface Identification and Casting

For the EEE surfaces, a volume is formed between the edge *e* and the emitter polygon (see Figure 4-14(a)). The grid traversal is extended to trace volumes bounded by planes, so that only the objects that are in the voxels that contain the planes or voxels cut by the planes are processed. The objects in these voxels then comprise a candidate list significantly smaller than the number of objects (Figure 4-14(b)). As an example (shown in Table 4-2), for scenes with 372, 696 and 1272 edges, the average number of candidate edges is 54.14, 65.83 and 78.40 respectively.

After rejecting edges clearly outside the volume, and those parallel to edge *e*, the number of edges *k* actually examined is quite small. From the statistics, the number *k* is 7.6 for an environment of 372 edges, and 7.23 for an environment of 1272 edges. The number of EEE surfaces that need to be examined is k^2.

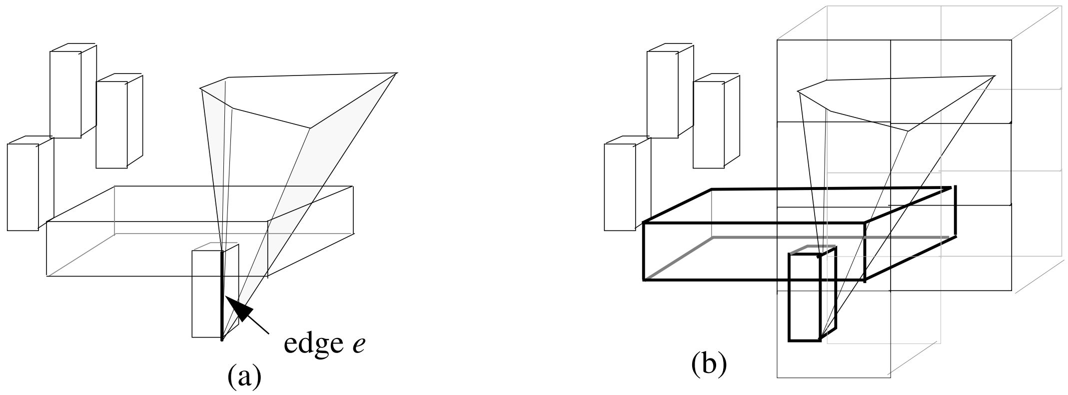


Figure 4-14. The EEE-Volume and the Grid

When casting the EEE surface, the intersection of the valid region of the quadric surface with the boundaries of the grid is found. The resulting quadratic curves are used to form bounding boxes on the maximal faces of the grid, that are then used to form a bounding box of the surface. Within this box, only the objects contained in the voxels that are cut by the quadric surface are examined, thus significantly reducing the number of objects actually intersected with the surface, as well as the resulting number of segments for which visibility processing is required.

4.4 An Incremental Algorithm for Computing the Backprojections

Each penumbral face has a unique backprojection. Thus it suffices to create one structure per face which contains the appropriate lists of emitter vertices and edge pairs. The backprojection instance can then be evaluated at a small cost at any point. In contrast, explicit computation (such as that by Nishita and Nakamae [42]) of the backprojection instance for each point in the face is expensive. It is equivalent to the computation of the umbra contour on the emitter, due to a point light source located at a point in the face.

To perform the *explicit* computation of the backprojection instance at a point P, the EV surface algorithm is used. Specifically, for all edges e lying between P and the emitter, the EV wedges are formed and cast. Edges are only inserted on the plane of the emitter. This creates the correct list of polygons of the backprojection instance on the emitter plane.

The information required to compute the backprojection in a face of the mesh, given the backprojection in a neighbouring face, can be efficiently stored with the edges of the mesh while the discontinuity surfaces are being cast. The backprojection of one face differs from the backprojection of its neighbour only by a small number of changes. If the features that define the discontinuity surface causing the edge connecting the two faces are stored with that edge, these limited changes can be performed incrementally. Thus, the expensive visibility calculations for the explicit computation are not required to calculate the backprojections in each face.

The incremental backprojection algorithm starts from a face with a known backprojection, then visits each face by crossing an edge from a neighbouring face. While crossing the edge to enter a face, the appropriate information is extracted and the incremental update is performed, resulting in the calculation of the backprojection for that face.

4.4.1 Classification of Mesh Regions

Once the mesh has been computed, every face in the environment is classified as a region of one of two types: a light region or a potential penumbra. The potential penumbral faces will either be penumbrae that have a non-null backprojection associated with them, or umbrae faces that are completely occluded from the current emitter.

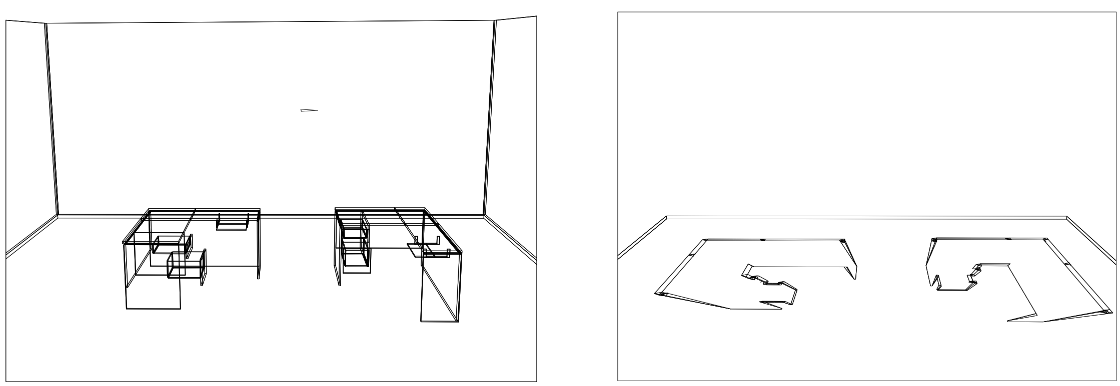


Figure 4-15. Penumbral Region Groups

Penumbral faces are connected together in groups. If it is necessary to traverse a light face to go from one penumbral face to another, the two faces are not part of the same group.These groups are bound by edge cycles of maximal edges, or edges that are part of the original polygon. In Figure 4-15 for example, each desk generates a disjoint region of umbra and penumbra, with respect to the triangular emitter.

All discontinuity mesh curves can be classified in two categories:

- EV induced lines, caused the intersection of an EV wedge and the scene polygon,
- EEE induced curves that are caused by an EEE surface intersecting the scene polygon.

4.4.2 An Incremental Backprojection Calculation Algorithm

The idea of an incremental algorithm is similar to that described by Gigus, Canny and Seidel [26]. The size of backprojections is in general smaller than that of view aspects, and for the purposes of rendering images, it is preferable to explicitly store the complete backprojection in each face, as opposed to the more involved priority search tree used in their view retrieval algorithm.

As described in the beginning of this section, the incremental algorithm performs a small number of updates to the backprojection when crossing an edge. Depending on the type of edge crossed, different actions are taken. The actions are similar to the visual event catalogue described by Gigus and Malik [25]. In our algorithm however, it is necessary to distinguish between invariant emitter vertices and other vertices, resulting in the need for modifications to that approach.

The incremental algorithm performs a modified depth-first search of the discontinuity mesh on every receiver surface. Starting from a face in light, for which the "backprojection" is the source itself, every face in each penumbral group is visited. When crossing an edge from one face to another, the appropriate incremental actions are taken. The actions required can be distinguished according to the type of edge being crossed.

If all the mesh faces on a polygon are in penumbra, it is necessary to find the backprojection in one face explicitly, and then perform the incremental updates for all the remaining faces. The use of the voxel structure allows this step to be performed efficiently since only the objects actually between the face in penumbra and the source are visited to find the explicit backprojection. In the statistics presented in Section 4.5.4, the time reported includes these steps where necessary.

EV induced edges. An EV edge corresponds to the appearance or the disappearance of a vertex behind an edge. Although care has to be taken to distinguish between the case in which the vertex involved is an emitter vertex and when it is not, the action taken is that of addition or deletion of an emitter vertex into the backprojection, and all other necessary updates of adjacent edges and backprojection polygons.

EEE-induced curves. Similarly, traversing a EEE curve requires the addition or deletion of up to two scene edge/scene edge elements in the backprojection.

4.4.2.1 Details of the Incremental Algorithm Updates

The outline of the incremental backprojection algorithm was given above. It is instructive to examine the details of the actions performed when crossing an edge in the mesh and the information that is required to perform the update.

In what follows, the incremental steps when crossing mesh edges resulting from emitter-EV events are presented in detail, while the steps for other types are sketched more briefly. Certain issues pertaining to degenerate cases are briefly discussed.

Emitter-EV Induced Edges

It is convenient to make the distinction between edges induced by EV surfaces, for which the vertex is on the emitter, called VE-edges, and those for which the edge is on the emitter called EV-edges. This distinction is necessary since the updates required to the backprojections are different.

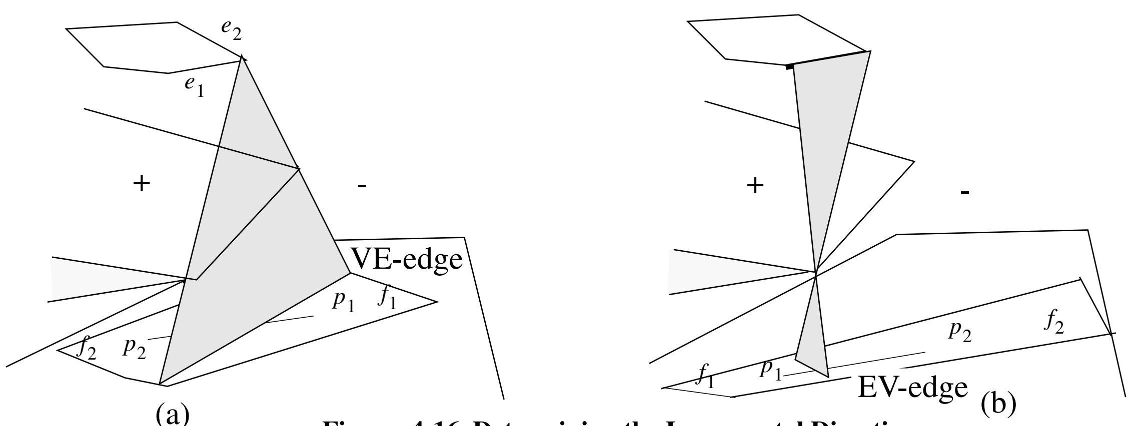


Figure 4-16. Determining the Incremental Direction

For both EV- and VE-edges between faces f_1 and f_2, it is necessary to determine whether features are inserted or deleted from the backprojection, when crossing from f_1 to f_2. This is performed by forming the wedge plane, using the correct sense of traversal of the edge with respect to the model. The representative interior points p_1 and p_2 lie on faces f_1 and f_2 respectively. If the value of the plane equation for point p_1 is positive and negative for p_2, an insertion is required, else a deletion is required (see Figure 4-16(a) for VE and Figure 4-16(b) for EV-edges).

In the case of insertion when traversing a VE-edge, the desired operation is the insertion of an emitter bp-element v into the backprojection. Let e_1, e_2 be the edges before and after the vertex v. Let also b_e be the blocker edge of the VE event. There are two possible cases:

1. The bp-element (e_1, b_e) or (e_2, b_e) already exists in the backprojection. In this case the bp-element must be replaced by (e_2, b_e) or (e_1, b_e) respectively. See Figure 4-17(a).
2. There is no bp-element containing b_e. Then a new list must be inserted into the backprojection with three bp-elements: (e_2, b_e) ; (v) ; (e_1, b_e) (see Figure 4-17(b)).

For deletion, the actions of Cases 1 and 2 must be reversed. Thus, it must be first determined if there is any bp-element that contains b_e, other than bp-elements (e_2, b_e) and (e_1, b_e). When the list of the backprojection containing bp-element (v) is of the form: (e_1, b_e) ; (v) ; (e_2, b_e), the list must be deleted from the backprojection, effecting the reverse action of Case 2. If not, the reverse of Case 1 must be performed; i.e. either the element (e_2, b_e) or the element (e_1, b_e) must be deleted, and (e_1, b_e) or (e_2, b_e) inserted respectively.

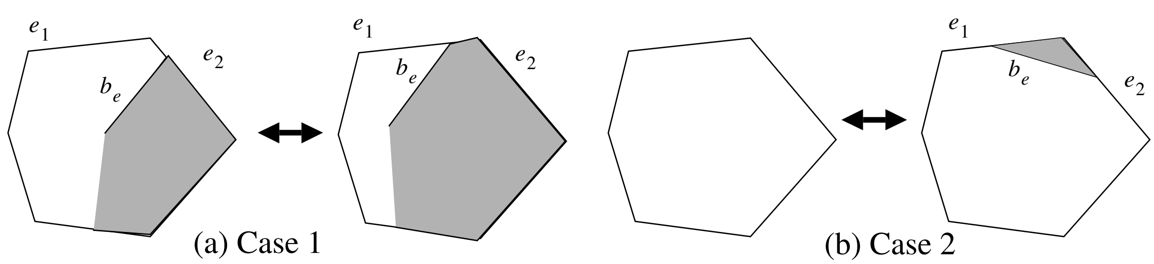


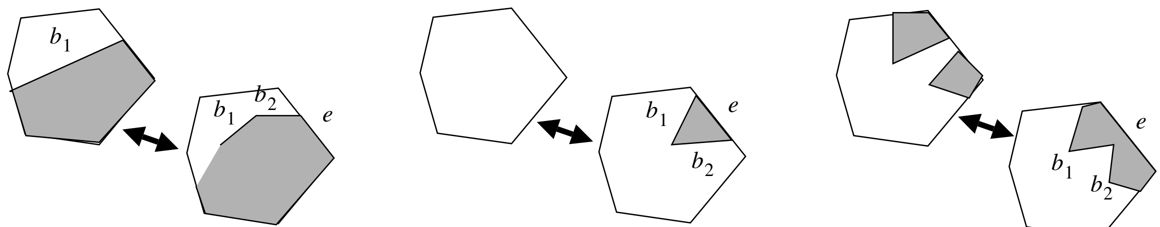
Figure 4-17. VE-Edge Incremental Updates

EV-edges are somewhat more involved. The incremental step when crossing such an edge involves the insertion (or deletion) of an scene edge/scene edge element. Let the blocker edges before and after the vertex v_b, be b_1 and b_2. Let the emitter edge of the EV-surface, be e. For the case of insertion, there are three possible configurations.

1. The bp-element (e, b_1) already exists in the backprojection, but not a bp-element containing b_2. Then (e, b_1) is deleted and (e, b_2) ; (b_1, b_2) inserted. This is shown in Figure 4-18(a). Corresponding actions are performed if (e, b_2) is in the backprojection.

2. No bp-element containing b_1 or b_2 exists in the backprojection. A new list is inserted into the backprojection containing (e, b_1) ;(b_1, b_2); (e, b_2) (see Figure 4-18(b)).

3. Bp-elements (e, b_1) and (e, b_2) exist in the backprojection. If they belong to different backprojection lists, the two bp-elements are deleted, and the two lists merged into one with the bp-element (b_1, b_2) at the joining point (see Figure 4-18(c)). Care must be taken to maintain the correct ordering of elements in the lists. If they belong to the same lists, the two bp-elements are deleted, and the bp-element (b_1, b_2) is inserted in that position.



(a) Case 1 (b) Case 2 (c) Case 3

Figure 4-18. EV-Curve Incremental Updates

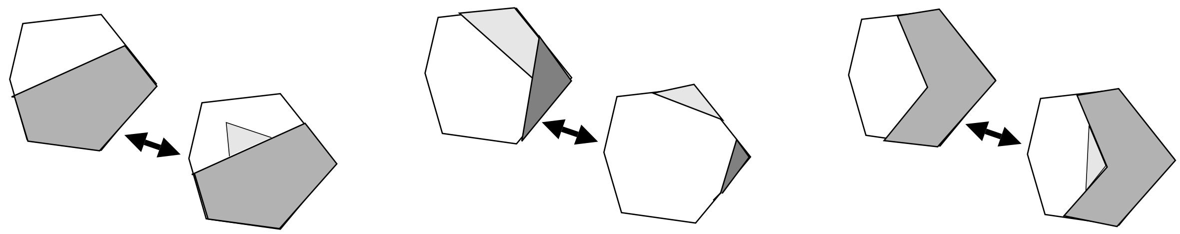
For deletion the corresponding cases must be identified, and the actions of Cases 1-3 reversed.The reverse of Case 1 occurs when a bp-element other than (e, b_1) exists that contains b_1 (b_2 respectively). Case 2 occurs when a list exists in the backprojection which consists only of (e, b_1) ; (b_1, b_2); (e, b_2) . Case 3 occurs in every other situation.

Other Curves

The details of the updates for the other events follow along the same lines as those given above for the emitter EV-curves. Illustrations and general descriptions of the configurations involved are listed below for completeness.

For non-emitter EV edges the action required is the addition or deletion of three scene edge/scene edge elements (see Figure 4-19(a)). For E_eEE curves the action involves the addition or deletion of a scene edge/scene edge bp-elements. The two edges involved however do not belong to the same blocker polygon (see Figure 4-19(b)). Finally for non-emitter EEE curves, the addition or deletion of two scene edge/scene edge elements is required (see Figure 4-19(c)).

When degenerate events occur, e.g. when an edge of an emitter is parallel to an edge of a blocker, one mesh edge actually corresponds to more than one change in the backprojection. In such cases, all the associated incremental changes must be made. In the example of the parallel edges, two emitter vertices would have to be inserted or deleted from the backprojection instead of one.



(a) Non-Emitter EV (b) E_eEE Curves (c) Non-Emitter EEE

Figure 4-19. Other Curve Incremental Updates

4.5 Results and Statistics

In this section running times and a collection of statistics are presented. Some of these numbers have already been referred to in the previous sections. Even though these statistics are not intended as conclusive evidence of the assumptions made, they are positive indications that for some interesting classes of scenes, the performance of our meshing algorithm can be expected to be good. Overall the algorithm running times and the statistics of scene characteristics indicate that the assumptions made hold true for the given set of test scenes. The class of scenes examined has not been formally defined, but the specific examples are drawn from office interiors.

The casting of emitter-EV surfaces, non-emitter EV identification and processing, and the calculation of E_eEE surface processing have been implemented and run on a set of scenes. The incremental backprojection calculation has also been implemented for EV- and E_eEE-induced mesh edges.

In the current implementation curved E_eEE-induced segments are approximated by line segments. In addition, only a partial implementation of the caching of the previous edge inserted has been implemented, and thus explicit searches are sometimes performed to locate the face in which a new edge should be inserted. The non-emitter EEE surface processing, and D^0 events have also not been implemented.

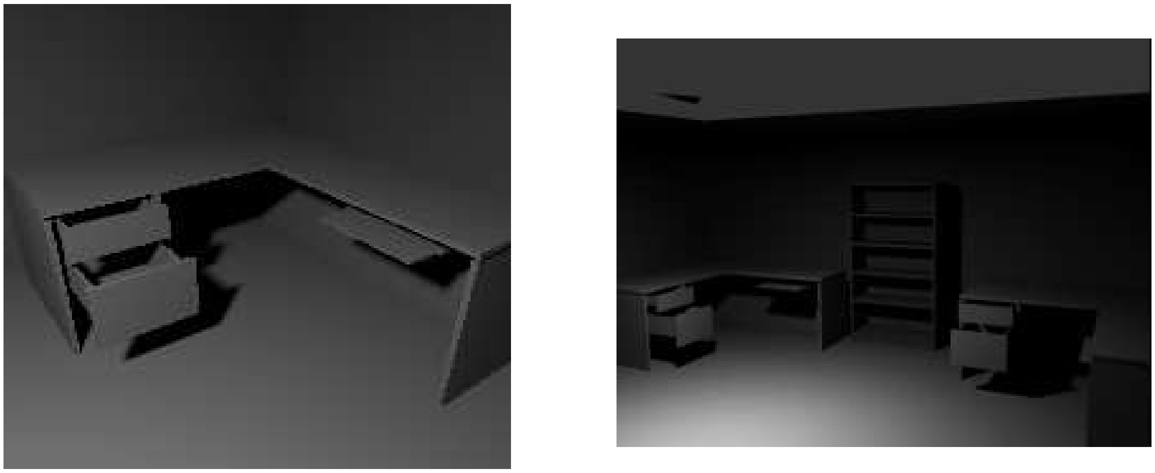


Figure 4-20. (a) One Desk Office and (b) Two Desks and Bookcase Exact Images

4.5.1 Complexity Analysis and Expected Running Time

A central expense of the algorithm is the casting of the discontinuity surfaces. For one such surface, the computational cost breaks down as follows. When casting a surface, l objects are visited. In the worst case l is $O(n)$ where n is the number of edges in the scene. The total number of segments resulting from the intersection of each discontinuity surface with each scene polygon is m, and the visibility processing costs m^2, resulting in $O(m^2)$ curves inserted. Each insertion costs p, where p is the number of faces searched to locate the face in which the edge is inserted. The quantity p is in the worst case $O(s)$, where s is the size of the mesh. Thus the cost of the casting of a single discontinuity surface and the resulting insertions is $l + m^2 p$, which is in the worst case $O(n + m^2 s)$. In the most pessimistic analysis, m is $O(n)$, and s is $O(n^6)$.

In average graphics environments, the use of the voxel grid restricts the number of objects l examined for each surface cast to be much smaller than n. The number m of segments for visibility processing is also a small number, since most often objects are distributed over the scene, rather than all grouped together. The number of objects cut by each EV-wedge tends to be small. This is especially true in scenes with a small source for which the width of the wedge is restricted. The average number p of faces visited when inserting an edge is also small due to the adjacency information stored in the face-edge-vertex structure. The number p is small also because the faces are arranged in groups formed as shadows of objects, rather than in long "chains" of faces, that would result in potentially many crossings. Finally the edges being inserted into the mesh are by definition all closely related in size, and thus no single edge will straddle a large number of consecutive faces. Thus the quantities l, m and p are all small numbers, often independent of n. As a consequence, the cost $l + m^2 p$ of casting a surface in practice small, and possibly also independent of n, for some classes of scenes. Statistics supporting each of these assumptions are presented in what follows.

There are $O(n)$ emitter-EV surfaces and worst case $O(n^2)$ non-emitter EV surfaces. Similarly there are $O(n^3)$ EEE surfaces in the worst case. As a consequence, the worst case complexity of our algorithm is $O(n^3(n + m^2 s))$ [3]. In contrast the algorithm of Stewart and Ghali [56] has a complexity bound of $O((n^2 + Pn^2 + dn + s)\log n)$, where d is the number of discontinuity surfaces in the environment, and P is the number of peak vertices, defined as a vertex for which all adjacent edges point down when directed outward from the vertex. The algorithm achieves this bound by avoiding the enumeration of all the $O(n^3)$ surfaces.

It is difficult to construct a scene for which all the worst cases would occur. An environment for which a large number of important EEE surfaces would occur would include a large number of skew edges, and a very large light source so that most of the resulting EEE surfaces cut the source. In addition the objects containing these edges should be stacked one on top of the other, thus making l close to $O(n)$. It is difficult to predict the size s of the mesh in such cases, since small objects will have a small number of faces. This also means that the average search time p for insertion will always be low in such a scene.

In practice, the number of non-emitter EV and EEE surfaces appears to be significantly smaller than n, and by using the voxel structure the observed running time for their identification grows linearly with n. Statistics to support these claims are presented below. Thus, the running time of the algorithm, for a reasonable class of scenes, demonstrates close to $O(n)$ growth.

---

3. The $O(m^2)$ step can be easily reduced to $O(m \log m)$ by using a plane sweep, reducing the complexity estimate to $O(n^3(n + sm\log m))$

There are two main strengths of our algorithm that allow it to achieve such running times. First, it takes advantage of expected geometric structure, in that the number of non-emitter discontinuity surfaces is expected to be small, and that the average number of objects intersected by any discontinuity surface, and thus the number of segments for which visibility processing has to be performed is also small. Second, it is output sensitive, since the cost of inserting an edge into the mesh depends on the size of the resulting mesh.

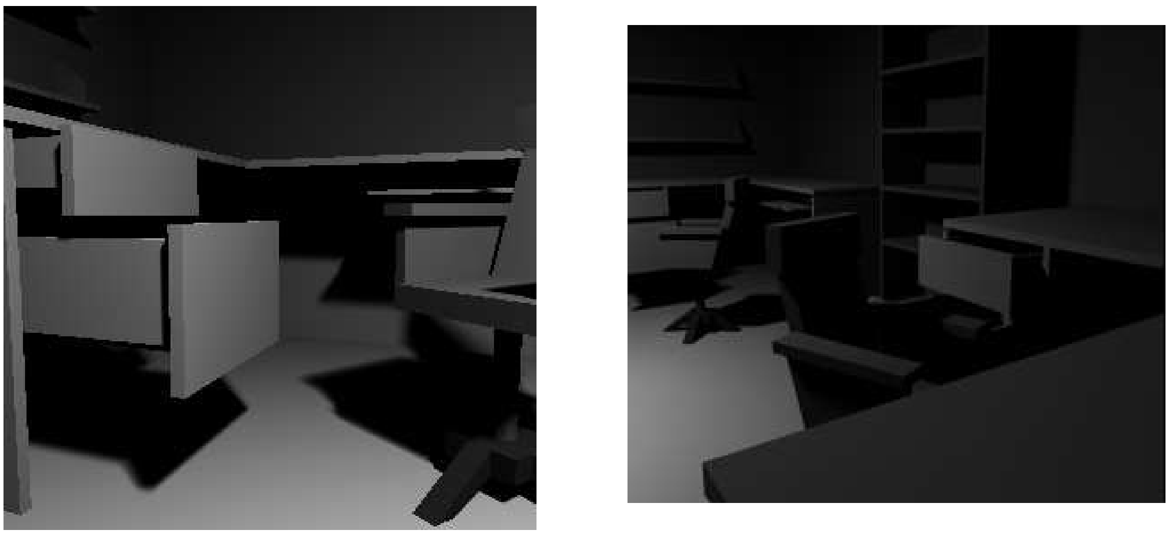


Figure 4-21. (a) 1 Desk and 1 Chair and (b) 2 Desks, 2 Chairs and Bookcase Scenes

4.5.2 Statistics for Emitter EV and E_eEE Implementation

Results are presented in Table 4-1 for scenes which consist of a table (Figure 4-3), one complicated desk (Figure 4-20(a)), two desks and a bookcase (seen in Figure 4-20(b) and in Appendix C as Colour Image 5(b)), one desk and one chair (shown as Figure 4-21(a) and Colour Image 6(a) in Appendix C), two desks and two chairs, and finally two desks, two chairs and a bookcase (shown as Figure 4-21(b) and Colour Image 6(b)). The most complicated scene (Figure 4-21(b)) contains 661 polygons and is a moderately complex office interior.

In Table 4-1, the fields *n*, e and *s* are the numbers of original (input) polygons, input edges and the size (number of faces) of the discontinuity mesh respectively. *Time (Grid)* is the time spent computing the mesh using the voxel spatial subdivision and *Time (NoGrid)* is the computation time without the use of the subdivision. The number *m* is the average number of line segments processed in the two-dimensional visibility step, and *max m* is the maximum number. *Avg. Objs.* is the average size of the candidate set found by scan converting the wedge on the grid (i.e. the set of objects the wedge actually tests for intersection). Finally p is the average number of faces traversed during the insertion of an edge.

Scene Name	*n*	*e*	*s*	*Time (Grid)*	*Time (NoGrid)*	*m*	*max m*	*Avg. Objs.*	*p*
Table	55	108	203	6.39	9.55	1.51	5	12.43	1.80
1 Desk	187	372	720	40.37	144.76	3.42	16	35.15	1.43
1 Desk & 1 Chair	331	636	1372	90.33	538.23	2.86	16	46.45	1.40
2 Desks	349	696	1437	86.21	672.07	3.39	29	39.32	1.41
2 Desks & Bookcase	409	816	1606	97.08	929.00	3.27	34	39.18	1.38
2 Desks & 2 Chairs	601	1152	2779	196.94	2395.96	2.92	29	52.66	1.45
2 Desks, 2 Chairs & Bookcase	661	1272	2944	208.22	2474.56	2.88	34	51.52	1.42

Table 4-1. Statistics for the Current Implementation of the Shadow Boundary Algorithm

Several important observations can be made. The overall computation time (*Time (Grid)*), including the non-emitter EV surface identification and processing, appears to grow close to linearly with the number of objects (or edges) in the scene.

From the results, it can also be seen that the two-dimensional visibility step is unimportant, since the average number of segments on the wedge plane is very small compared to the scene (m vs. n or e). In addition, the number of faces (p) intersected on the average during the insertion of each edge into the mesh is a small constant (between 1.40 and 1.80). This suggests that the incremental mesh building approach can be expected to perform much better than the plane sweep alternative. Also, it is interesting to note that the size of the mesh (s) does not seem to grow quadratically with the number of edges in the environment, but at a slower rate, much closer to linear.

Of particular interest is the fact that the average number of objects in the candidate queue for each wedge cast is small compared to the number of objects (*Avg. Objs.* vs. n). Depending on the scene configuration, the growth of *Avg. Objs.* does not necessarily depend on the increase in the size of the scene (for $n = 601$ and $n = 661$ *Avg. Objs.* is 52.66 and 51.52 respectively). The number of objects visited is central in the complexity of the algorithm. If this number does not grow with the number of objects, then the cost of casting a discontinuity surface does not grow as the scene becomes more complex.

When *Avg. Objs.* is high the scene is said to have high *visibility complexity.* As observed in the tests, when objects were added to the specific office scene (chairs, bookcases etc.), the visibility complexity does not grow much, since these objects do not, for the great majority of light/object configurations, interfere with each other with respect to the light source. In some cases this number will decrease when additional objects are added (for example for 2 desks and 2 chairs, when the bookcase is added).

4.5.3 Statistics for Non-Emitter Events

In addition to the identification of non-emitter EV-surfaces, the identification step has been implemented for non-emitter EEE surfaces. This is a prototype implementation, to determine how many such events exist and to get a sense of the computational expense. More involved algorithms can definitely be used for the identification stages, resulting in smaller computational expense.

Scene name	*e*	*n.e. EV/k*	*n.e. EV Surfs.*	*EV Time*	*EEE Queue Len*	*n.e. EEE/k*	*n.e. EEE Surfs.*	*EEE Time*	*#Emitter Wedges*
1 Desk	372	25.59	79	18.59	54.14	7.60	0	25.59	1167
1 Desk & 1 Chair	636	31.56	168	37.23	66.85	6.73	0	52.98	2049
2 Desks	696	29.88	232	42.42	65.83	7.85	0	55.45	2232
2 Desks & Bookcase	816	31.68	232	38.37	72.38	8.55	0	50.59	2598
2 Desks, 2 Chairs	1152	38.52	437	80.57	83.41	7.55	3	107.5	3762
2 Desks, 2 Chairs & Bookcase	1272	36.75	437	83.99	78.40	7.23	3	112.05	4128
Scene Statistics with Larger Source									
1 Desk	372	23.64	633	19.96	54.49	7.38	116	28.72	1527
1 Desk & 1 Chair	636	32.91	2450	46.02	70.12	8.92	490	64.35	2712
2 Desks	696	30.25	1466	39.90	73.33	8.64	261	54.74	2948
2 Desks & Bookcase	816	29.33	1704	45.87	70.32	8.44	308	64.41	3462
2 Desks, 2 Chairs	1152	37.58	3787	88.62	87.77	8.91	753	124.16	4999
2 Desks, 2 Chairs & Bookcase	1272	36.29	4025	94.77	84.61	8.76	800	134.33	5513

Table 4-2. Non-Emitter Surface Statistics and Identification Times

Given these current restrictions, the statistics for all non-emitter surfaces are reported in Table 4-2, for all the test scenes presented in Table 4-1. The total number of non-emitter EV surfaces varies from 79 to 437 (6% and 10% of emitter EV surfaces). The average number of edges contained in the voxels of the EV-pyramidal volume (*n.e. EV/k*) varied from 25.5 to 36.7 for scenes of 108 to 1152 input edges (6.8% and 2.8% of input edges). The two scenes were modified to include a larger source, placed in such a way that more of these surfaces cut the emitter. These results are shown on the lower half of Table 4-1. For these new scenes the number of non-emitter EV surfaces was much larger, close to the number of emitter EV surfaces. However, average the number of edges in the volume did not change significantly, remaining between 2% and 6.5% of the input edges.

For non-emitter EEE surfaces, the average number of objects examined at each edge (*EEE Queue Len.*) varied from 54.1 to 78.4, while the average number of edges in the volume actually used to form EEE surfaces (*n.e. EEE/k*) was 7.2 and 8.5. In most cases there were no non-emitter EEE surfaces that cut the source, while there are 3 non-emitter EEE surfaces for scenes with two chairs. These surfaces are occluded and therefore do not contribute to the mesh, allowing the rendering of these scenes even though the complete non-emitter EEE surface processing has not yet been implemented.

For the scenes with the enlarged emitter, the average number of polygons examined again does not change much (54.4 to 84.6), while the number of edges in the volume varies from 7.3 to 8.9. The total number of EEE surfaces inserted into the candidate list to be cast is significantly larger varying from 116 to 800. Nevertheless, the number of these surfaces appears to grow close to linearly with the number of objects.

The cost of identifying all the non-emitter surfaces for the scenes with the small source is close to double that of the computation time for the emitter EV surfaces. It is important to note however that this computation time grows close to linearly in n. For example for the one desk, one chair scene the total identification time is 90.21 seconds and for two desks, two chairs scene it is 188.07 seconds. Similar relationships can be seen for all the scenes tested, including those with the larger source in which the number of non-emitter surfaces increases significantly. The identification step is thus not expected to become overwhelming with increased scene complexity, assuming the visibility complexity does not grow significantly. Much faster identification computations are expected with a more complete implementation.

4.5.4 Results for the Incremental Algorithm

The incremental backprojection calculation algorithm has been implemented for simple scenes, in which only emitter EV and E_eEE surfaces exist. Due to this restriction, only simple environments can be tested. Results from two tests are reported. In the first, the environment consists of one receiver parallelepiped, a smaller blocker parallelepiped and a triangular light source. In the second, a table with four legs and one wall and a floor is used. The results of the tests are shown in Table 4-3.

Scene Name	*n*	*s*	*Mesh Time*	*Time(s) Explicit*	*Time(s) Incremental*
One Blocker	19	154	0.77	2.06	0.16
Table Scene	35	94	2.61	9.19	0.57

Table 4-3. Statistics for the Incremental Backprojection Algorithm

In Table 4-3, the time to explicitly compute the backprojections in every face (**Time(s) Explicit**) is compared to the time taken to compute the backprojections using the incremental algorithm (**Time(s) Incremental**). For both the One Blocker and the Table scene, the speedup is significant, from 2.06 seconds to 0.16 seconds and from 9.19 seconds to 0.57 seconds respectively.

Even better results should be expected when the scene is more complex and thus the explicit computations are much more expensive. This is particularly true since for larger scenes (e.g. the desk scenes mentioned above), the average size of the backprojections is small (3-10 edges), while the scene is much larger (hundreds of objects).

4.5.5 Exact Image Computation

To generate an image containing an area light source, previous methods (Campbell and Fussell [9] and Lischinski, Tampieri and Greenberg [39]) explicitly compute the visible portion of the source at the vertices of the mesh. Interpolants are then constructed using these values. Consequently, the resulting images are generated with approximations of the real radiance values in the penumbra, rather than exact values. In addition, the reported cost of explicitly computing the illumination at the vertices, in these algorithms, is equal or larger than that of the mesh computation.

Given the abstract representation of the backprojection in every penumbral face, it is possible to compute exact images of penumbral regions at reasonable cost. Such images have only been produced for either very simple environments or at large computational cost (Nishita and Nakamae [42]) in the past. Specifically, the radiance value at any pixel can be computed exactly by finding the instance of the backprojection at the visible point, and by applying Eq. (2-26) for each fixed polygon, without the need to query the environment to determine the visible portion of the source. This allows multiple views of scenes with area sources to be computed exactly, at very low cost. Examples of such images are shown in Figures 4-3, 4-20 and 4-21 (Figures 4-20 and 4-21 are shown as Colour Images 5 and 6 in Appendix C). These images are extremely useful for reference when attempting to evaluate approximation techniques such as those described in the following chapter.

When the incremental backprojection algorithm is used, the time to compute the illumination in the mesh is drastically reduced. This is because previous methods required the explicit computation of the backprojection at the vertices of the mesh, resulting in computation times costing as much as or more than the mesh computation time. In contrast, the new algorithm allows all necessary illumination computations to be performed in a fraction of the time to compute the mesh. This renders the new approach even more usable for the exact computation of images with penumbra.

4.6 Summary and Conclusions

In this chapter a complete and efficient discontinuity meshing algorithm has been presented. In addition an incremental backprojection calculation algorithm has been introduced. The current implementations of both the meshing and the backprojection calculation algorithms have been presented, that indicate that the complete discontinuity mesh can be computed in reasonable running time, and that the use of the incremental backprojection algorithm results in substantial savings in the computation of radiance in the penumbra.

The implementation of our algorithm is the first that has been used to compute the complete discontinuity mesh for a set of interesting scenes. The only other algorithm capable of computing the complete mesh is that of Stewart and Ghali [56] that has been proposed concurrently with this work. The results of an implementation are not yet available for this algorithm, and thus comparison of results is not possible.

EV surfaces are treated with an improved and complete extension of Heckbert's algorithm [33]. E_eEE surfaces are treated using a simple and efficient post-process, while a new general algorithm is presented for non-emitter EEE surfaces. The new EEE surface algorithm is the first meshing algorithm in which the details are given of the computation of quadratic curves on the polygons resulting from the quadric EEE surfaces, and the visibility computation along such surfaces for these curves.

For all EV and EEE surfaces, extensive use is made of a uniform, voxel-based spatial subdivision structure. This allows only a small portion of the environment to be visited when casting the discontinuity surfaces, resulting in significantly accelerated computation times.

By identifying key geometric structure properties of a class of interesting computer graphics environments and subsequently taking advantage of these properties, test data suggests that our algorithm achieves close to $O(n)$ running time, where n is number of edges, for moderately complex scenes. Four such properties particularly effect the successful running times, despite pessimistic asymptotic complexity bounds. First, the number of objects that need be visited to determine the intersections of a discontinuity surface with the environment is significantly smaller than the total number of objects and does not necessarily grow as the number of objects increases. The algorithm uses the voxel grid spatial subdivision to capitalise on this fact. Second, the number of objects cut by each discontinuity surface cast into the environment is a very small number and also does not necessarily grow with the number of objects. Third, identifying the EV and EEE surfaces that do not contain emitter features can be performed in close to linear running time, again using the voxel grid. Fourth, the structure of the mesh faces is such that the new incremental insertion algorithm allows edges to be inserted in the mesh at minimal expense.

Once the complete mesh has been computed and the backprojections calculated in each face, exact values of radiance for every pixel in the penumbra can be computed at very low cost. The ability to compute images exactly with area sources in reasonable time is extremely important and has not been possible before. The rendering of such analytic images is also important for the evaluation of approximation algorithms for scenes with partial occlusion, such as that described in the following chapter.

The incremental backprojection algorithm allows the backprojection to be computed in every face of the penumbra efficiently, in time $O(s)$, where s is the size of the mesh. This drastically reduces the expense of computing the illumination, once the mesh has been computed. In previous work, this computation has been performed explicitly at all the vertices of the mesh where radiance values were desired (e.g., Campbell and Fussell [9], Lischinski, Tampieri and Greenberg [38][39]). In all these approaches, this computation equalled or surpassed that of computing the mesh. Thus by using the incremental backprojection algorithm, the computation of illumination for scenes with areas sources is drastically reduced, with the additional benefit that the images computed are exact at every pixel and not interpolated approximations such as those computed using the previous approaches.

Chapter 5

Structured Sampling for Scenes with Occlusion

5.1 Reconstructing Illumination in the Penumbra

The sampling and reconstruction of radiance within penumbral regions was first attempted by Campbell and Fussell in [9] and later by Lischinski et al. [38] (described in detail by Tampieri in [60]). Campbell and Fussell recognised that the variation of illumination in the penumbra was complicated and difficult to predict. They thus used numerical optimisation techniques to determine the behaviour of radiance in regions of shadow. In the approach by Tampieri, and by Lischinski et al., the discontinuity mesh computed is incomplete. Even though a number of discontinuities are found, the function of illumination within the faces of their mesh contains discontinuities of derivatives, making it difficult to reason about the structure of radiance within the mesh faces.

The properties of the radiance function in the penumbra can be more clearly understood using the complete mesh with backprojection information, described in Chapter 4, and using the analysis developed for the unoccluded case in Chapter 3. The research we report in this chapter is based on understanding gained from, and synthesis of, the previous work. The accomplishment of this chapter is thus twofold: to better understand the properties of radiance in environments with occlusion, and to construct compact and accurate low order polynomial interpolants that represent radiance in scenes with occlusion.

To understand the properties of illumination within penumbrae, empirical and numerical studies have been performed. Following this study, the construction of interpolants is separated into two main components: the determination of the domains on which the interpolants are defined, and the construction of the interpolants themselves. New algorithms for reducing the geometric complexity of the mesh, as well as the use of varying degree interpolants are presented. The interpolants for radiance in the penumbra are then combined with the interpolants built for unoccluded regions, in the manner described in Chapter 3. Continuity and domain shape issues are discussed in some detail. Two algorithms for rendering the scenes for which the interpolants have been built are then presented. Finally, numerical tests are run using the new algorithms that demonstrate the improvements achieved from their use.

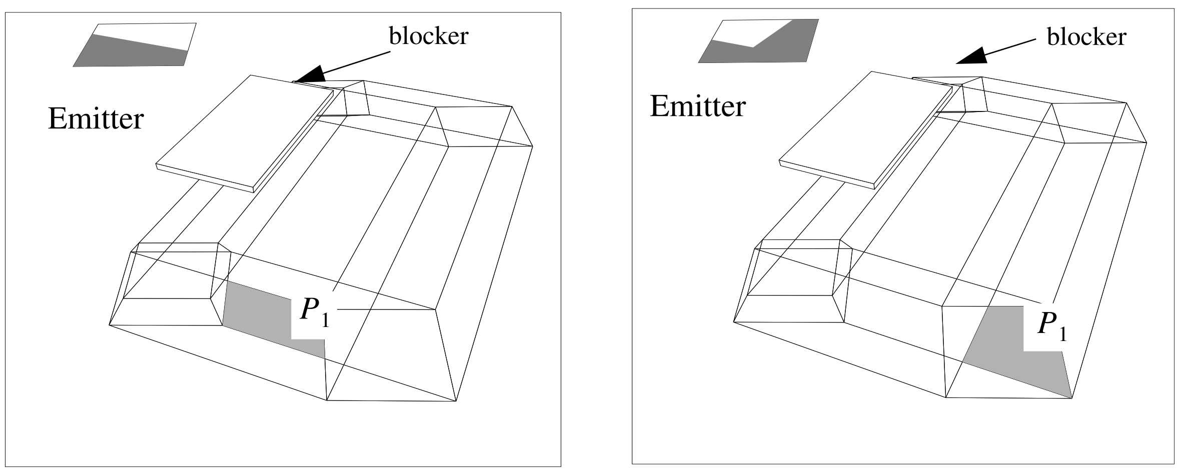
The complete structured sampling algorithm constructs a piecewise polynomial interpolant representation of radiance for general environments. The resulting mesh has several desirable properties: it is a sparse representation, due to the use of the unoccluded algorithm in regions of light and the mesh reduction algorithm for penumbral regions. The interpolants are also of low degree: linear or quadratic or mixed. These are both desirable properties for the finite element formulation of the global illumination problem. A higher-order finite element algorithm, that uses this mesh as an initial approximation, is presented in detail in Chapter 6.

5.2 Properties of Radiance Functions in Penumbral Regions

Campbell and Fussell [9] were the first to observe that radiance in a penumbral region can exhibit multiple minima and maxima. Numerical optimisation was used to determine these critical points. Tampieri [60] and Lischinski et al. [38], segmented the penumbral domain by the mesh generated solely from emitter-EV events. They subsequently postulated that within each face or cell of this mesh the radiance does not vary much. A subsequent adaptive subdivision step was nonetheless used when large radiance discrepancies were observed.

In what follows, an extensive empirical study of radiance in the penumbra of several scenes indicates that when the complete mesh is computed, the radiance does appear to be well behaved within each face of the mesh. For the majority of cases, radiance is monotonic along any direction within a face, and varies slowly. In these cases, the radiance is sufficiently well behaved to be represented by quadratic interpolants.

If radiance in a face is monotonic, good error bounds can be guaranteed for the interpolants that are built even without the computation of derivatives. There are some cases in which radiance in the penumbra is not monotonic, and some cases in which it varies substantially and can thus not be well captured by quadratic interpolants. We call the latter case radiance with *rapid variation*. These configurations are not common, but are nonetheless quite evident and are analysed below.



(a) (b)

Figure 5-1. Convex and Concave Backprojection Instances

5.2.1 Definitions and Scene Properties

Before examining the behaviour of radiance in detail, a set of definitions and basic properties of scenes with occlusion is presented. This is not intended to be a definitive list of properties. It is the result of empirical and numerical studies of radiance in the penumbrae, performed over the course of this research.

The computation of the complete mesh and the backprojection representation of the σ-aspect in each face give a significant advantage over previous work in the area, since the computation of exact radiance values can be performed extremely efficiently, once the complete mesh is constructed. All the graphs of radiance and its derivatives presented in this chapter were generated from the testbed system used to implement and test the ideas of this research.

Backprojection Properties

Consider the example Figure 5-1(a), in which a view of an environment with one blocker, one receiver (the floor) and a rectangular source is shown. For any point on the shaded face of the mesh, the backprojection corresponds to a fixed polygon. The shaded region on the rectangular emitter shows the visible portion of the emitter at point P_1. The radiance at P_1 can be considered the value of the radiance function of this polygonal source. This is an example of an *instance* of the backprojection, as defined in Section 4.2.1.

Definition 5-1: *The backprojection of a face is **convex** or **concave**, if every instance of the back-projection in that face is itself a convex or concave polygonal source, respectively.*

An example of a convex backprojection is shown in Figure 5-1(a) and a concave example is shown in Figure 5-1(b).

Definition 5-2: *A backprojection of a face is **disconnected**, if every instance of the backprojection in that face consists of more than one polygon, while it is **simple** if any instance consists of only one polygon.*

An example of a disconnected backprojection is shown in Figure 5-2(a), while the backprojections in Figure 5-1(a) and (b) are both simple.

Faces with concave backprojections can be completely surrounded by EV-induced edges, caused by the interaction of a source edge and a shadow vertex (corner) of a blocker. As a consequence, if the source edge is long, the face will be quite large. The instances of the backprojection can be considered concave emitters that can cause multiple extrema as observed in Section 3.1.1. In addition, the problem is exacerbated by the fact that radiance for every point in a face is caused by a different concave emitter. Because the faces are large, functions with multiple extrema interact, and can cause minima and maxima of radiance along different directions within a face.

Extremal Analysis from a Constant Source

Consider a scene that is illuminated by a polygonal source, and in which the complete discontinuity mesh of that source has been computed. Consider also any line within the penumbra, and the radiance as it varies along this line. For lines in the penumbra on which the shape and size of the backprojection instances do not change significantly, the behaviour of radiance can be analysed in exactly the same fashion as for the unoccluded case in Chapter 3.

Along such lines, the maximum and inflection points can be determined and the function can be split into monotonic, and then convex and concave regions. For example, the edge shown as a thick line in Figure 5-2(b) contains the maximum of the radiance function from the polygonal source defined by the backprojection instance, shown shaded on the rectangular source. This maximum can be seen in the radiance plot in the inset rectangle on the right.

A interesting way of characterising the behaviour of radiance on a line with respect to a constant polygonal source is the angle formed by the two lines joining the maximum of the radiance due to the source and the edge endpoints. An example of this angle is shown by the dashed lines in Figure 5-2(b).

Definition 5-3: *The angle formed by the endpoints of a line segment in the penumbra on a receiver and the maximum point on the same receiver due to an invariant source, is called the* ***edge-source angle****.*

When this angle is small, the assumption that the extension of the line segment passes through the maximum of radiance due to the (almost invariant) source, can be considered approximately correct, and thus the segment does not contain the maximum. If the angle is large, the line segment probably contains the maximum of the radiance function due to the constant source.

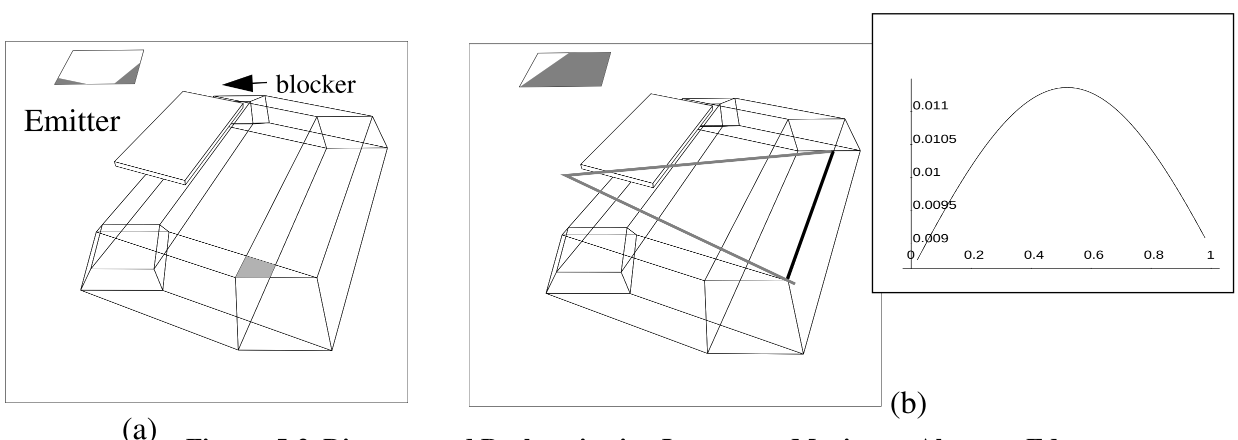


Figure 5-2. Disconnected Backprojection Instance, a Maximum Along an Edge

Singularities

As noted by Lischinski et al. [38] and Tampieri [60], radiance along a surface is singular at points at which two surfaces touch. Thus at a vertex in the mesh joining faces of umbra, penumbra and light, the radiance value is defined as a limit depending on the direction from which the vertex is approached. A singular vertex is shown in Figure 5-15(b).

5.2.2 Characterisation of Rapid Variation and Non-Monotonic Radiance Behaviour

From the tests run, several prevalent factors emerged that affect the behaviour of radiance in the penumbra and create local extrema or rapid variation of the radiance function. The *first* is the nature of the backprojection in the corresponding face. The *second* is the relative position of the maximum and critical regions of the constant or almost constant backprojection instances. The *third* is the interaction with the unoccluded illumination. The *fourth* is the relative size, position and orientation of the source-blocker-receiver triple, which becomes problematic in cases in which large faces are created in the mesh.

The study of radiance properties in the penumbra is driven by the need to construct interpolating functions. In what follows we list radiance behaviours that cause naive construction of interpolants to fail. This list is not exhaustive, since it is based on empirical and numerical studies (see also Section 5.2.3). It is however encouraging that the trends observed in suites of simple scenes in which several parameters were varied carry across to the more complex environments tested.

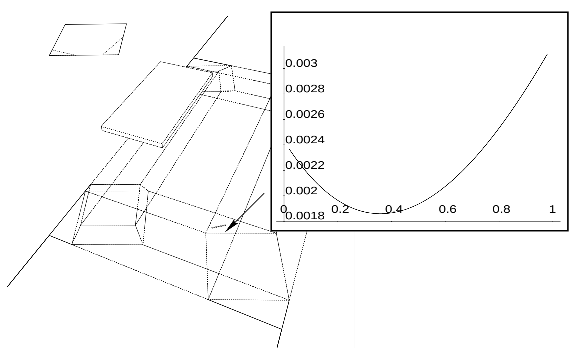


Figure 5-3. Extrema Due to Disconnected BackProjections

Disconnected or Concave Backprojections

If the backprojection on a face (or along an edge) is not simple, it is likely that there exists a line in the face or along an edge for which the radiance will display one or more maxima. This phenomenon is caused by the fact that illumination due to a disconnected backprojection results in instances containing n polygons. This is completely equivalent to illumination from n separate unoccluded light sources, that may have up to n maxima. Non-monotonic behaviour in this case may cause "troughs" in the radiance function, as shown in Figure 5-3, from the overlap of two "tail" regions of a disconnected backprojection with instances containing two polygons. In this figure and those that follow, additional lines can be seen in the mesh that are due to the segmentation of the receiver into light and penumbral regions (see Section 5.5.1). These lines correspond to the bounding box of a penumbral group and should be ignored in the discussion concerning the properties of radiance in the penumbra.

Since concave backprojections degenerate into disconnected backprojections when an EV-induced edge in the mesh is crossed, similar non-monotonic behaviour may occur in faces or along mesh edges that have concave backprojections.

Disconnected and concave backprojections usually occur in faces that are small in area. This is the case since faces with such backprojections are typically bounded by edges of the mesh caused by the interaction of source edges (EV-induced edges) with scene vertices, or by curves caused by E_eEE events. Both mesh curves tend to be smaller than VE-induced edges, for environments in which the source is on average smaller than the blocker objects.

Identifying concave and disconnected edges in the mesh is trivial once the backprojections for each face have been computed, either explicitly or using the incremental algorithm. Once these edges have been identified, a simple monotonicity test can be applied. Details are given below in Section 5.2.3.

Lines Containing Maxima of Illumination Functions

As mentioned earlier, there are lines in regions of the penumbra with simple backprojections, along which the area and shape of the backprojection instance polygon does not vary significantly. For the VE-edges in the mesh, the backprojection instance polygon remains constant, and thus radiance behaviour can be analysed exactly as for the unoccluded case.

If the edge-source angle of the edge is large, the maximum caused by the backprojection instance polygon may cause the radiance to display non-monotonic behaviour. If the edge-source angle is small, the extension of this line will pass close to the maximum, and therefore the radiance along the line can be characterised by analysing the relative position of the line with respect to the maximum and inflection points. This case will always result in the appearance of a local maximum rather than a local minimum (see Figure 5-2(b)).

Calculating the maximum for the backprojection instance would be prohibitively expensive. As an alternative, the maximum of the unoccluded source, which has already been computed, is used to calculate the edge-source angle. Since long edges that have a large edge-source angle are typically VE-induced edges, the backprojection instances in question are similar in shape to the original unoccluded source, and are thus the locations of the corresponding maxima are closely related. Since only an indication of the angle size is required, this approximation is expected to be sufficient.

For any edge, the edge-source angle is less than 180 degrees, and therefore its size can be estimated from the dot product of the two defining lines. If the angle is greater than a given tolerance, the radiance along that line should be checked for monotonicity.

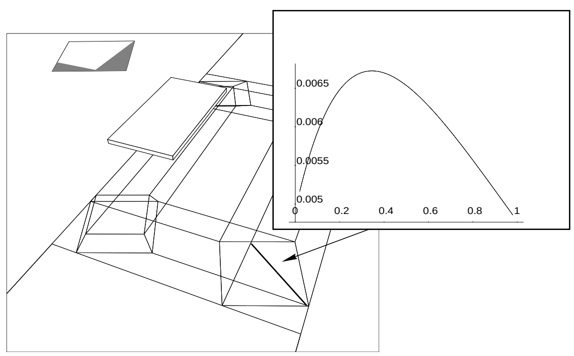
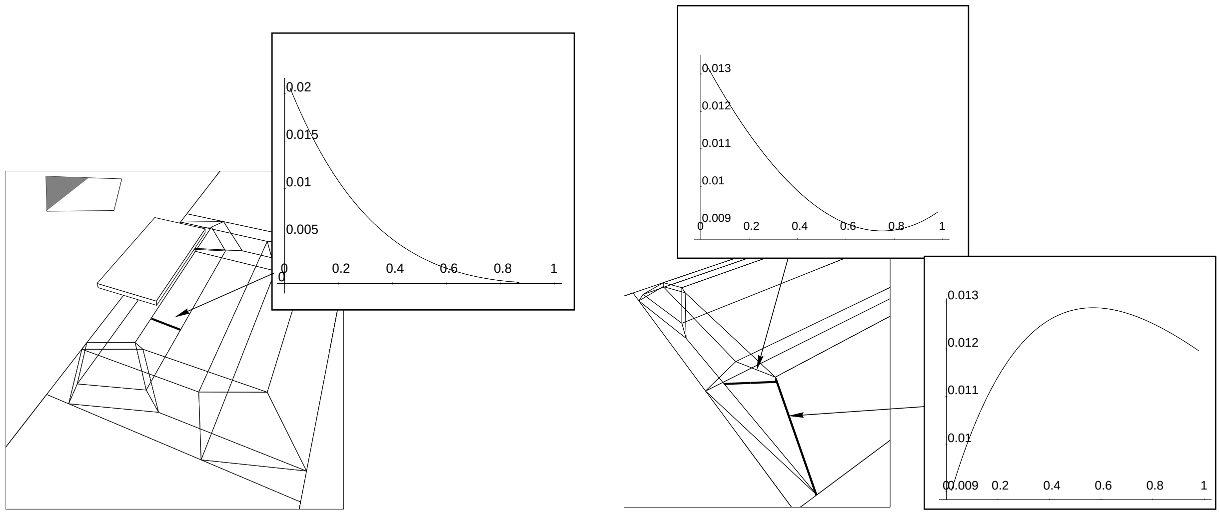


Figure 5-4. Interaction with Unoccluded Regions

Edges Ending in Light Regions

Consider a line in the mesh that does not lie on the external penumbral boundary between light and shadow but has however one endpoint in light. An example is shown as a thick line in the mesh of Figure 5-4. The visible area of the source generally increases along this line, going from penumbra to light, as a smaller portion of the source becomes occluded. As a consequence the radiance on a line is generally increasing as the region of light is approached.

The unoccluded illumination along this line may be increasing or decreasing. If the unoccluded illumination is decreasing, the radiance in the penumbra, which is an increasing function, will tend to smoothen as it approaches the unoccluded regions. Again, this can result in non-monotonic behaviour, although it is necessarily a local maximum. An example of this is shown in Figure 5-4.



(a) (b)

Figure 5-5. Fast Varying Radiance in the Penumbra, and Opposite Extrema Within a Face

Large Faces with Fast-Varying Radiance and Opposite Extrema

When a source edge is large with respect to a blocker, large faces can be created in the mesh. The size of these faces will depend also on the distance of the source from the blocker and of the blocker to the receiver, and on their relative orientation.

In such faces, it is possible that the radiance function will display rapid variation (see Figure 5-5(a)). This depends on the maximum and minimum size of the backprojection instances in these faces. The worst case is that of long edges that meet at a sharp corner on a source. Such an example was encountered in the tests performed, and the radiance variation became too fast to be represented by a quadratic when the edge of the source had approximately the same size as the edges of the blocker.

Faces with concave or disconnected backprojections, particularly in the presence of light source edges that are very long, can cause the existence of opposite extrema (i.e. a minimum and a maximum) along different directions in a face. An example is shown in Figure 5-5(b).

5.2.3 Implementing Non-Monotonic Edge Identification and Statistics

A set of test programs has been written that analyse the behaviour of radiance in a set of scenes. Tests for four sequences of simple scenes are computed as well as two scenes with more complex geometries.

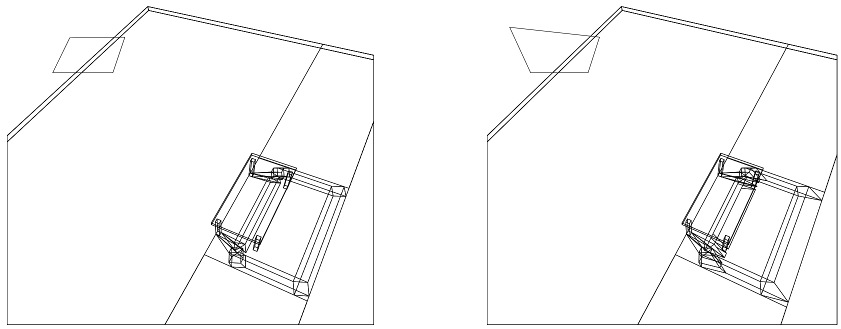


Figure 5-6. Two Frames of the "Grow" Sequence

The four simple sequences involve a quadrilateral source illuminating a simple table with four legs, and consist of five frames per sequence. In the first sequence, the source starts as a square, and subsequently has one vertex move away to become elongated (Grow sequence). The geometry of two frames of this sequence are shown in Figure 5-6. In the second sequence the square source is lowered so that it comes closer to the blocker (Down sequence), in the third it is placed at the corner of the table (Corner sequence) and moves down. In the last sequence the square source moves away while maintaining the same height (Away sequence). The complicated scenes are a complex desk model in a room, and the same model with the addition of a chair and shelves on the wall. The geometry of the Desk scene is shown in Figure 5-21(a) and that of the Desk & Chair scene in Figure 5-22(a).

For the purposes of gathering these statistics, the full monotonicity check calculates the derivative at twenty points along each edge. If the derivative has a zero, radiance along the edge is classified as non-monotonic, else it is classified monotonic. The edges examined are the actual edges of the discontinuity mesh, and the edges of the triangulation (see Section 5.3.3), that are interior to the original mesh faces.

This computation is very expensive and cannot be used as part of a structured sampling algorithm, since it would multiply the cost of creating the interpolants in the penumbra by a large factor.

Instead we use a three-point algorithm, in which the two endpoints and the midpoint are tested to determine whether or not the radiance is monotonic across the line.

In Tables 5-1 to 5-3, the statistics of these tests are shown. For the four sequences, average values over the whole sequence are reported, while for the Desk and Desk & Chair scene a single case was run.

Scene	Edges	N.M.	%N.M	Light	Disconn.	Concave	EeEE	Other	Angle
Grow (avg.)	602	57	9.59%	11	21	3	17	4	23.57
Down (avg.)	571	53	9.42%	11	19	4	14	4	26.63
Corner (avg)	615	55	9.01%	10	19	3	18	2	0.33
Away (avg.)	599	51	8.54%	12	18	2	13	4	3.91
Desk	1939	70	3.61%	15	20	11	19	5	0.00
Desk & Chair	3717	153	4.12%	32	40	31	41	9	0.00

Table 5-1. Mesh Edge Monotonicity

In Table 5-1, the statistics for edges of the mesh are shown. In the column **Edges** the total number of edges is shown, **N.M**. is the number of non-monotonic edges, with the percentage on the total shown as **%N.M**. The columns **Light**, **Disconn**., **Concave**, **E_eEE** correspond to edges that neighbour regions of light, neighbour faces with disconnected and concave backprojections, or edges that are E_eEE-induced, respectively. The column **Other** shows the number of edges that are not in the above categories, while **Angle** shows the average edge-source angle of these edges.

Scene	Faces	N.M.	%N.M	Light	Disconn.	Concave	Other
Grow (avg.)	804	62	11.07%	15	36	10	21
Down (avg.)	760	57	10.55%	16	31	9	18
Corner (avg.)	817	59	11.06%	18	33	8	28
Away (avg.)	798	56	9.77%	16	31	9	17
Desk	2367	90	5.45%	40	26	24	34
Desk & Chair	4588	193	5.86%	79	54	60	67

Table 5-2. Interior Edge Monotonicity Statistics

Statistics for the edges of the triangles that are interior to faces of the mesh are reported in Table 5-2. In the column **Face** the total number of faces is shown, **N.M**. is the number of non-monotonic faces, with the percentage on the total shown as **%N.M**. The columns **Light**, **Disconn**., **Concave**, correspond to faces that neighbour regions of light, or to faces that have a disconnected and concave backprojection respectively. The column **Other** shows the number of non-monotonic faces that are not in the above categories.

Finally, in Table 5-3, statistics are shown giving the success rate of the three point monotonicity test. The columns **Tests**, **Success** and **Fail** show the total number of tests, the number of times the three-point test succeeded and failed respectively. The percentage of failure is shown in **%Fail**.

Scene	**Tests**	**Success**	**Fail**	**%Fail**	**Light**	**Disconn.**	**Concave**	**Other**
Grow	3645	3288	357	10.86%	28	96	28	10
Down	3425	3102	323	10.41%	36	82	31	18
Corner	3742	3369	373	11.07%	35	98	24	34
Away	3562	3260	302	9.26%	41	81	24	14
Desk	1576	1428	148	10.36%	28	16	18	10
Desk & Chair	3449	3070	379	12.35%	57	35	42	22

Table 5-3. Statistics for the Three-Point Monotonicity Test

The above statistics clearly show that the percentage of non-monotonic edges is small overall (between 4 and 10%). A large number of the non-monotonic edges are in or adjacent to faces with disconnected backprojections, which were observed to be very small in size in the tests run. In addition, the E_eEE "edges" are the line approximations of what in reality are curves, and thus radiance may actually be discontinuous, giving the false impression of non-monotonicity. Notice how the percentage of non-monotonic faces is smaller for the desk and the desk and chair scene. This is due in part to the fact that the light source for these tests is smaller compared to that of the test suites.

It is also clear that the three-point monotonicity test performs quite well overall, succeeding for nearly 90% of the tests it performed. The simpler test fails mainly for faces with disconnected backprojections, and for faces neighbouring light regions. In practice it can be used with significant confidence.

5.3 Interpolant Domain Construction for Penumbral Regions

The problem of constructing an interpolant for radiance in a general scene can be split into the construction of interpolants for regions that are unoccluded, and into the construction of interpolants for regions in shadow. The former is treated using an extension of the methods of Chapter 3, described in Section 5.5, while the determination of interpolants for penumbral regions is presented next. As was mentioned earlier, there are two aspects to the interpolant definition: the determination of the domains on which the interpolants are defined, and the choice of basis functions and their degree. In this section, the determination of domains is presented.

5.3.1 Continuity and Geometry Constraints for Interpolant Domains

The complete discontinuity mesh, and the accompanying backprojection information, computed using the methods presented in Chapter 4, is a natural segmentation of the radiance function over a surface. Edges in this mesh are discontinuities either of value, or of first or second derivative. Characterisations of these discontinuities can be found in Heckbert and Winget [35], Heckbert [34], Lischinski et al. [38] and Tampieri [60].

Value discontinuities occur only where objects touch, and therefore the boundary of such regions delineates areas in which the radiance function has value zero, because they are completely hidden from the light. These discontinuities do not have to be treated, since the function can be considered defined only in the regions of non-zero radiance.

Discontinuities of second derivative (or first derivative when degenerate events occur), need to be treated since they occur in regions of penumbra. These discontinuities constitute the majority of lines in the discontinuity mesh.

The geometric nature of the faces, or regions in which the σ-aspect is the same, is complicated. It includes highly irregular regions that can be small, concave and with small angles between edges. This mesh constrains the construction of interpolant domains, since some of these discontinuity lines need to be maintained to achieve high quality reconstruction of radiance.

The relative importance of the discontinuities is difficult to assess without significant numerical computation. Some of the second derivative “jumps” can be small, while others can be quite large (e.g., Figure 5-7(a) compared to Figure 5-7(b)). In some cases the effect on the actual radiance function is very evident while in others the effect is negligible (Figure 5-7(b)). In addition, radiance in very small regions cannot display extremely large differences in value, because the shape of the polygons in the backprojection instances cannot change much, and neither can the form-factors that determine the value of radiance at any point in a face.

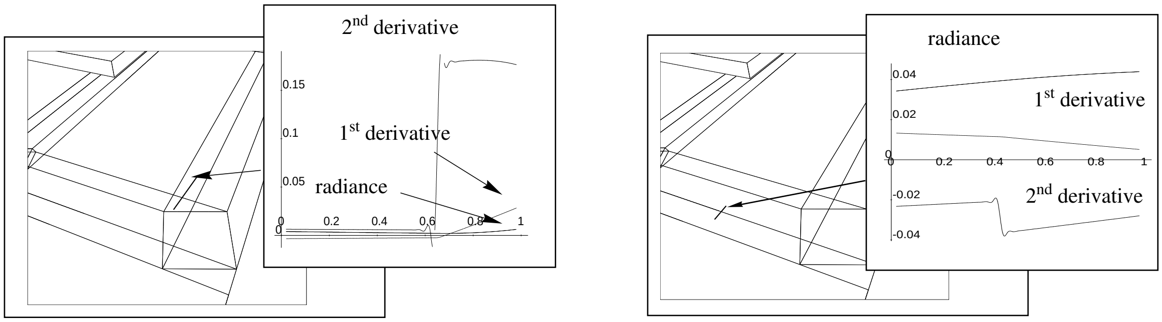


Figure 5-7. Effect of Discontinuities

To accommodate the highly irregular nature of the faces in the mesh, triangles are selected as the domains over which to construct interpolants. Specifically, a *proper* triangulation of the penumbral domain is performed. A triangulation $\tau = \{T_o, T_1, \ldots, T_n\}$ into the triangles T_i is proper if each pair of triangles intersect at a vertex, a complete side or not at all, and the union of all triangles equals the domain (Prenter [44]).

The basis functions used (presented in more detail in Section 5.4.2) are linear, quadratic or mixed linear/quadratic Lagrange interpolating polynomials. Overall, due to the natural decomposition into regions of smooth radiance, further subdivision of the faces beyond the initial triangulation does not seem to be required very often. The situations for which additional subdivision is required will be outlined in Section 5.3.5, and an algorithm to identify this need and perform the subdivision will be presented.

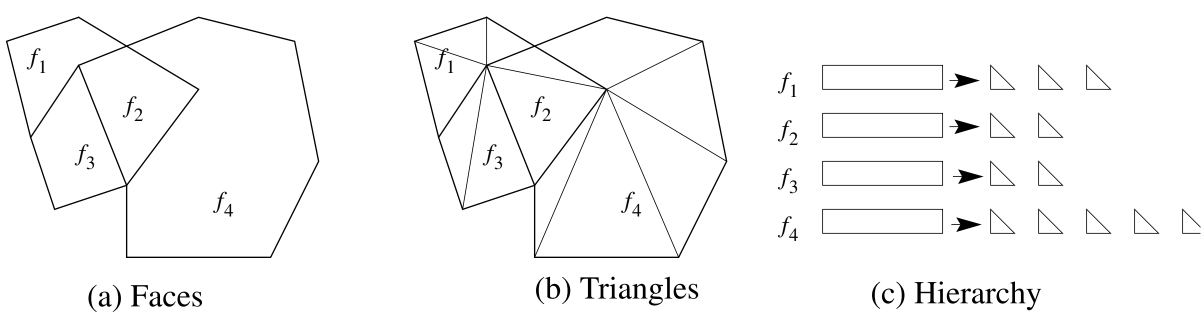


Figure 5-8. Face/Triangle Hierarchy

5.3.2 Data Structures

The segmentation of the penumbral region into the faces of the discontinuity mesh defines a natural data structure in which adjacency information is stored. Details of this structure were presented in Chapter 4. In addition each face contains the backprojection corresponding to that face.

The triangulation of each face is stored as a second level in a hierarchy (see Figure 5-8). Maintaining this hierarchy is important for two reasons. First, it allows the exact solution to be calculated efficiently, without requiring the ray-casting procedure to query the larger number of triangles. Second and most importantly, it distinguishes amongst edges of the triangulation that are true discontinuities (edges in the mesh) and edges that are simply triangle boundaries, and over which radiance and its derivatives are smooth.

The face structure maintains the detailed adjacency and backprojection information that is common to all the contained triangles. This information is used in rendering and whenever the mesh needs to be manipulated.

The values of radiance at the vertices are stored along with the vertices in the complete discontinuity mesh, in the same manner as Lischinski et al. [38]. Thus these values are shared between neighbouring faces, and no redundant computation or storage is required for radiance values that are shared amongst different interpolants.

5.3.3 Triangulation of Faces

The mesh faces in the penumbral regions are sometimes concave irregular polygons. To allow triangulation of these regions, two separate approaches are used.

For polygons with one concavity, the concave vertex is connected to all other vertices, resulting in a triangulated face (Figure 5-9(a)). This type of face accounts for a large proportion of all concave faces.

For more complicated concave regions, a general algorithm is used, based on a public domain program by Schumacher and Seum [51]. The concave region is split using y-spans, to create trapezoids. The resulting trapezoids are split into triangles across the diagonal. This construction is shown in Figure 5-9(b).

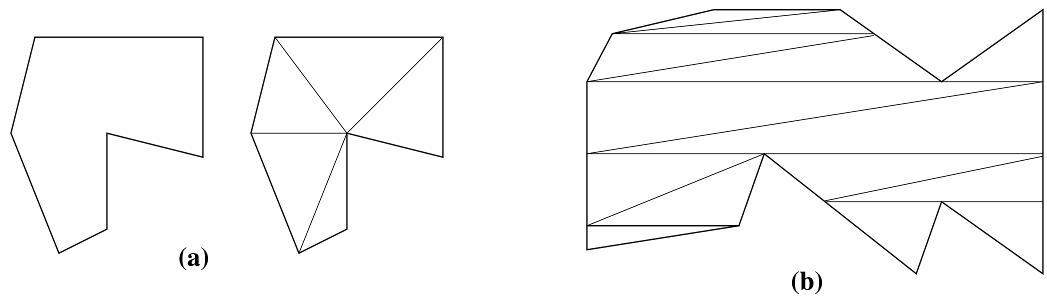


Figure 5-9. Triangulation for Concave Faces

For convex polygons, a simple triangulation scheme is used. All the vertices are connected to the centroid (Figure 5-10(a)). This avoids the creation of sliver or irregular triangles, at the expense of introducing an extra vertex.

The triangulation of the concave regions may result in cases in which the triangulation of the penumbral region is not proper (Figure 5-10(b)). To avoid this, the triangle on which the geometric discontinuity occurs is subdivided into as many triangles as needed to ensure continuity. This subdivision is shown with the dashed lines in Figure 5-10(b).

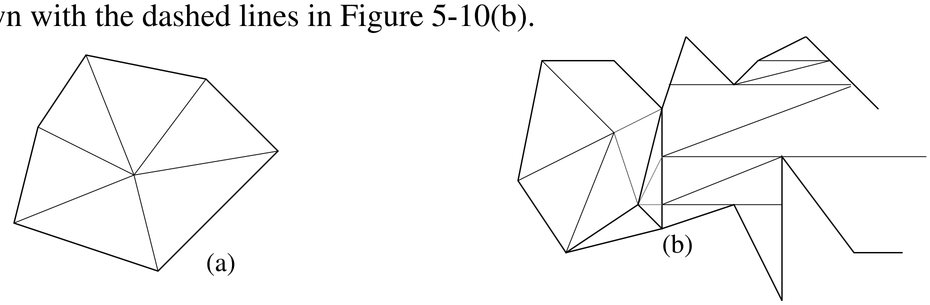


Figure 5-10. Triangulation for Convex Faces and Continuity Enforcement

5.3.4 A Mesh Reduction Algorithm

As mentioned above, faces of the discontinuity mesh can be arbitrarily small and may also have edges with high aspect ratios. In addition, the triangulation of concave regions can result in triangles that are very small or that have small angles.

For the construction of piecewise smooth interpolant domains, it is desirable to reduce the number of such triangles as much as possible. Larger triangles, and triangles with larger angles are more stable numerically, and in addition provide the benefit of a better theoretical error estimate. Specifically, consider a six-point quadratic interpolant s_N on a triangle, and the radiance function f. The max norm in an interval $[a, b]$ is defined as follows:

$$\|f(x)\| = \max_{a \le x \le b} |f(x)| \quad . \tag{5-1}$$

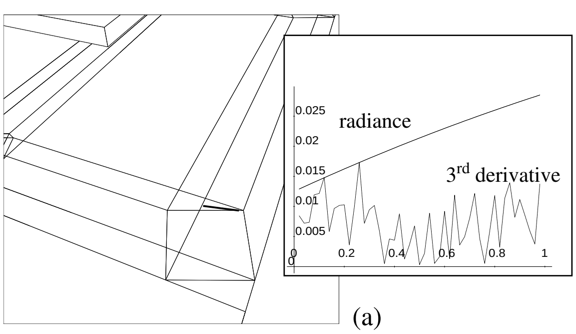
Following Prenter [44], the error bound with respect to the max norm for interpolant s_N is:

$$\|f - s_N\| \le \frac{8M_3}{\sin\theta} h^3, \tag{5-2}$$

where h is the longest edge of the triangle, and θ is the smallest angle of the triangle. The constant M_3 is equal to the maximum value of the first, second and third derivative of f within the triangle. In general:

$$M_n = \max\{\|D^1 f\|, \|D^2 f\|, \ldots, \|D^n f\|\}, \tag{5-3}$$

where $D^i f$ is the i'th derivative of f. In the case of unoccluded illumination, and for some of the faces in the penumbra, the magnitude of the third derivative is many times larger than that of the radiance itself (see Figure 5-11(b)), rendering Eq. (5-2) somewhat meaningless. However, there are other faces in the mesh in which the third derivative is small, and thus the size of the smallest angle θ can play an important role in the quality of the approximation achieved. In Figure 5-11(a) the absolute value of the third derivative is smaller than the radiance value. The oscillations are the result of numerical instability in the computation of the third derivative. The magnitude of the third derivative is however consistently less than the magnitude of the radiance function.



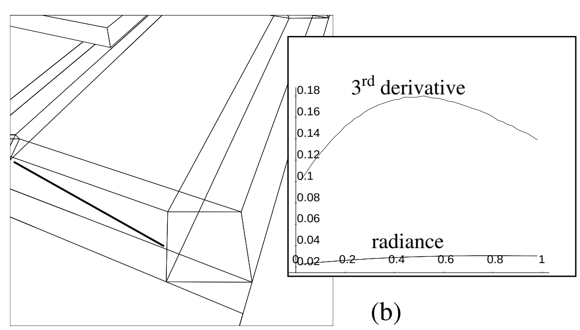


Figure 5-11. Absolute Values of Third Derivative in the Mesh Faces

5.3.4.1 An Area-Based Edge Elimination Algorithm

For the reasons outlined above it is desirable to eliminate all overly small faces in the mesh. In addition, a reduction in the size of the mesh is beneficial in general. A smaller mesh allows the radiance function to be represented with a smaller number of triangular interpolants. This allows faster rendering and is also important when such a representation is used for light transfer simulation (see Chapter 6). In the results it shall be shown that a large mesh reduction with negligible error is achievable.

For a given penumbral group, the algorithm first determines the face with the largest area. The guiding strategy is to then remove all faces that have area less than `tol%` of the largest face.

Data Structures

When eliminating edges from the mesh, corresponding faces are deleted. It is therefore necessary to maintain the backprojection information, together with the geometry of the deleted faces in the resulting merged face. This information can be discarded after the construction of the interpolants if the exact solution is no longer desired. If however the ability to render the exact solution is still required, the complete backprojection information can be maintained. As an example consider Figure 5-12, in which three edges are removed, resulting in the removal of faces f_2 and f_3. The resulting data structure for the larger merged face f_1 is shown in Figure 5-12(b), where the dashed line represent the edges that have been removed.

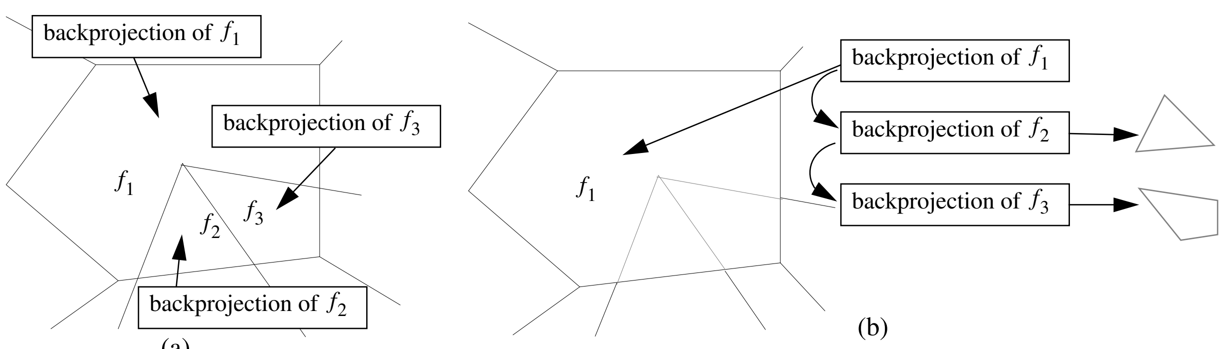


Figure 5-12. Data Structures for Edge Elimination

When removing edges from the mesh, configurations occur in which edges contain vertices that previously marked the point of intersection with now deleted edges. Such a vertex splits an edge, but is not connected to anything. These vertices are removed from the mesh, and all adjacent data structures are appropriately updated.

Eliminating Concavities

The original face f_1 shown in Figure 5-12(a) is a concave polygon. It is often the case that the tip of a smaller face will enter a larger face, creating a concavity in the polygon (for example in Figure 5-12(a) the tip of f_2 or f_3 causes f_1 to be concave). For the reasons outlined above, it is desirable to eliminate such faces.

Each concave face is visited, and the vertices for which the two edges on the face form an angle greater than 180 degrees are identified. Edges are removed from the concave face, if the corresponding small face has area below the tolerance. If all the faces around the vertex can be removed, the final dangly edge is also removed from the mesh.

Eliminating Small Faces

After the first set of concave faces has been treated, a number of small faces that have area less than the maximum may still remain. For each such face the following procedure is applied:

```
remove_small_face( Face f )
{
    Edge e;
    sort all edges e of f by area of neighbouring face
    for each edge e in sorted list do
          if can_remove( e ) then
                 remove_edge( e );
                 return;
          endif
    done
}
```

In the procedure `can_remove ( e )` determines whether the removal of the edge is possible. The following rules are applied:

1. No edge of the boundary between umbra and penumbra is removed.
2. No edge of the boundary between umbra and light is removed.
3. If the removal of an edge results in a convex face becoming concave, it will not be removed.

After this procedure is applied, all remaining dangly edges are removed from the remaining face. After an edge is deleted, the tail and head vertex are searched to determine if the edge extended into the neighbouring faces. If it did, the neighbouring edges are removed, if such a removal does not violate rules 1-3.

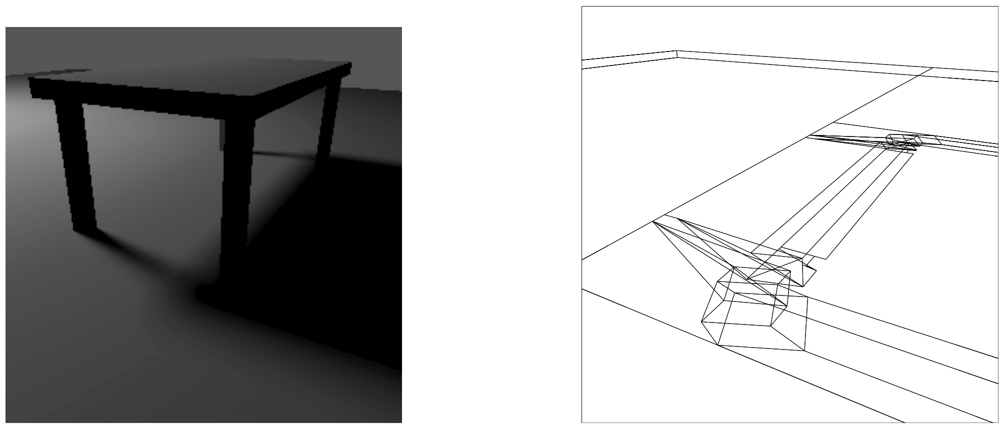


Figure 5-13. Image and Mesh Before Reduction

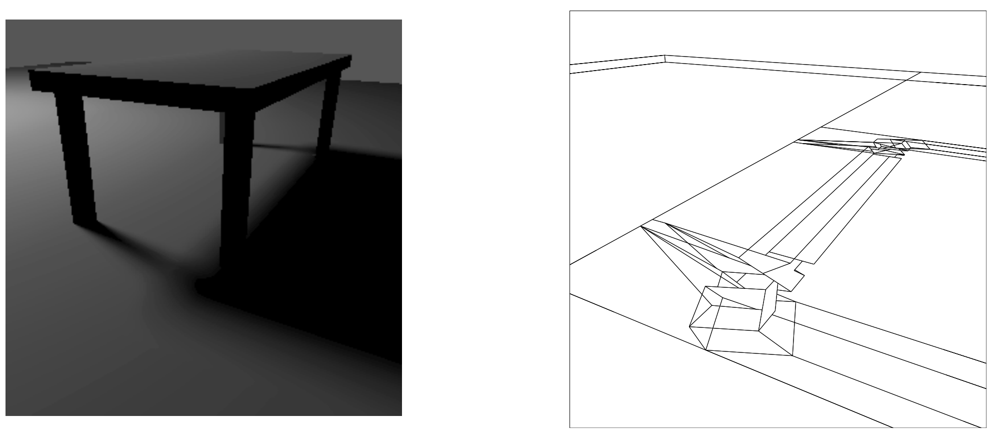


Figure 5-14. Image and Mesh After Reduction

The removal process (concavity and small face removal) is repeated until all faces that are smaller than the tolerance are removed, or until no more faces can be removed.

In Figure 5-13 an close up of a table scene is shown using the standard algorithm before mesh reduction, while in Figure 5-14, the same scene and mesh are shown after reduction. Notice how the mesh has been has been reduced to 46% of the original faces before reduction, but the quality of interpolation is still quite high. The rendered images of Figures 5-13 and 5-14 can be found in Appendix C as Colour Images 7(a) and 8(a).

5.3.5 Adaptive Subdivision

As mentioned earlier, the subdivision effected by the discontinuity mesh appears often sufficiently fine for the construction of interpolants. As noted by Tampieri [60], adaptive subdivision can be required in large faces in which the radiance varies rapidly. As mentioned above, large faces are typically bounded by VE-edges and EV-edges, due to long source edges interacting with long edges of blockers, or corners of a blocker. In the former case, the variation of radiance may be steep, but it will generally be monotonic (Figure 5-5(a)). In the latter case, simple quadratic interpolation will be insufficient when radiance displays a maximum in one direction and a minimum in another (Figure 5-5(b)). This occurs typically in faces with disconnected or concave backprojections.

Rapidly Varying Radiance

Faces with rapidly varying radiance will be large compared to other faces in the penumbral group, and will display a large maximum difference of radiance. When a large face (i.e., in the top 5% of a penumbral group in terms of face size) is encountered, which has edges with a large difference in radiance values, the adaptive subdivision algorithm for the tails (Section 3.4.2.2) is applied along these edges. Derivative evaluations are expensive, so the maximum number of steps is limited even if the tolerance is not met.

Opposite Extrema Within a Face

The three-point monotonicity algorithm is applied in the faces with concave or disconnected backprojections as described below (Section 5.4.1). The monotonicity test is first applied along the edges of the triangles in the original face. If the maximum difference in radiance is greater than a given tolerance, it is also applied along the remaining diagonals of the face. If opposite extrema are encountered (i.e., a maximum on one direction and a minimum in another), adaptive subdivision is performed on that face.

The adaptive subdivision algorithm subdivides the face along lines formed by the positions at which the opposite extrema were located, thus splitting the face into regions where opposite extrema are less likely to occur. A subsequent test is applied in the newly formed regions, and it is repeated until a maximum subdivision depth is encountered, or no opposite extrema can be located. This process is illustrated in Figure 5-15(a).

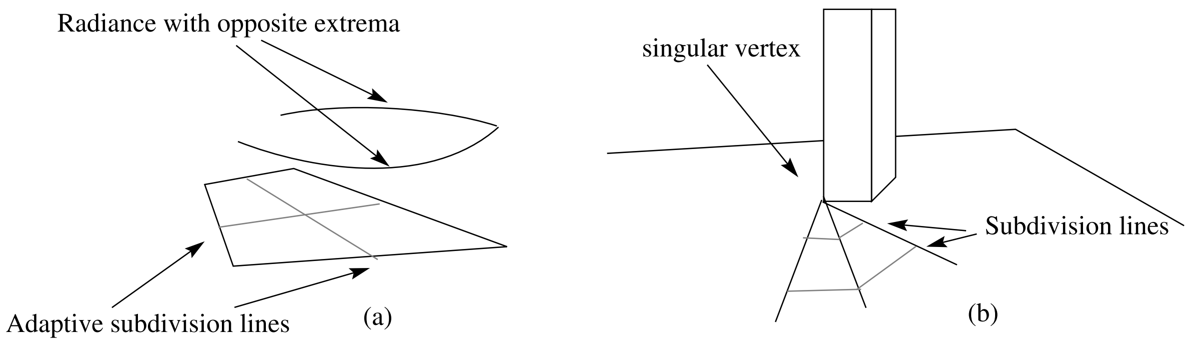


Figure 5-15. Adaptive Subdivision Algorithms

Singular Vertices

Singular vertices are multi-valued, depending on the direction of approach. A special purpose interpolation algorithm is devised to address this problem, and is described in Section 5.4.3. This treatment is not always sufficient, since the interpolation performed using this algorithm may still be prone to error. To address this problem the radiance along edges that join at the singularity is analysed, and subdivision is performed where appropriate.

For each edge leading to the singularity (see Figure 5-15(b)), the derivatives along that edge are computed in the neighbourhood of the singularity and at the other endpoint. The algorithm for subdivision in the unoccluded tail regions is applied along this edge, until the maximum error incurred by the interpolant falls below the preset tolerance. This leads to subdivision around the area of the singularity.

5.4 Constructing the Interpolants for Penumbral Regions

Once the triangular domains have been constructed, we have to choose appropriate basis functions and the calculate the coefficients to construct the interpolant. The empirical and statistical analysis presented in Section 5.2 suggests that for many cases in the penumbra, linear interpolation of radiance is sufficient. Thus the interpolants constructed by our algorithm are linear, quadratic or mixed.

Because the complete mesh has been computed, the umbral regions are well defined. On these domains, constant basis functions are used with a value of zero. For this step, the computation of the complete mesh is a necessity, since boundaries of the umbra are often (EEE-induced) curved edges, which were not computed by previous algorithms (e.g., Lischinski et al. [39]).

To achieve the construction of mixed linear/quadratic interpolants, it is first necessary to characterise the edges of the mesh and the edges interior to the faces with a required degree of interpolation. Each edge or face receives a characterisation of linear or quadratic, and the appropriate basis function is assigned to the triangle, maintaining continuity.

5.4.1 Selecting a Degree for the Edges of the Triangular Domains

Once the set of triangular domains has been selected, the degree of the interpolant must be such that the resulting interpolant approximates radiance in a satisfactory manner and thus allows the generation of high quality images.

Ideally, the radiance function would be analysed using radiance derivatives and curvature along the edges of the triangular domains, and it would be decided if a linear interpolant is sufficient to capture the variation along that line. Such computation is unacceptably expensive however, since the numerical computation of the derivatives would more than double the cost of computing the coefficient values for the basis functions.

As an approximation, we develop cheap, approximate criteria for choosing the degree required on an edge in the mesh or a triangle edge in the interior of a face. The criteria are based on whether or not radiance along an edge is monotonic, and whether the difference between radiance values is larger than a preset tolerance in pixel RGB values.

For mesh edges, the type of backprojection of adjacent faces is first taken into account. As can be seen from the statistics in Section 5.2.3, a large proportion of non-monotonic edges are adjacent to faces with disconnected and concave backprojections. Edges induced by E_eEE events, edges neighbouring a light face, and finally VE-induced edges that have a large edge-source angle can also be non-monotonic.

For triangular edges interior to faces, similar properties hold. Specifically, faces with concave or disconnected backprojections, as well as faces neighbouring light regions, often display non-monotonic behaviour.

To classify radiance behaviour along the edges, the three-point monotonicity test is applied. For monotonic edges, the difference between the radiance values at the endpoints is computed. This difference is converted into integer RGB values, and if the difference between radiance values at is less than `lin_tol`, the edge is classified as linear.

For non-monotonic edges a test is applied between edge endpoints and the midpoint radiance values. If both differences are smaller than `lin_tol`, the edge is classified as linear.

```
construct_penumbral_interpolants( Mesh m, degree_tol )
{
    for each edge e of mesh m do
          select_degree_on_edge( e, degree_tol )
    done
    for each face f of mesh m do
          Triangles T = get_triangles( f )
          for each triangle t in T do
                 for each edge et of t interior to f do
                       select_degree_on_inter_edge( et, degree_tol );
                 done
          done
    done
    for each face f of mesh m do
          Triangles T = get_triangles( f )
          for each triangle t in T do
                choose_and_build_appropriate_basis_function( t )
          done
    done
}
```

Figure 5-16. Basis Function Selection and Construction Algorithm

Once each edge of the triangular domains has been classified, a triangular basis function is selected and the coefficients calculated based on the triangle coordinates and the radiance values at the appropriate points. The interpolant construction algorithm can be summarised in the pseudo-code of Figure 5-16. In Figure 5-17 the same scene as that of Figure 5-13 is rendered after the application of the degree selection algorithm. Notice the quality of interpolation is still high despite the fact that 50.7% of the interpolants are linear. The dashed lines show the edges of the mesh that have been assigned linear interpolants. The image of Figure 5-17 can be found in Appendix C as Colour Image 9(a).

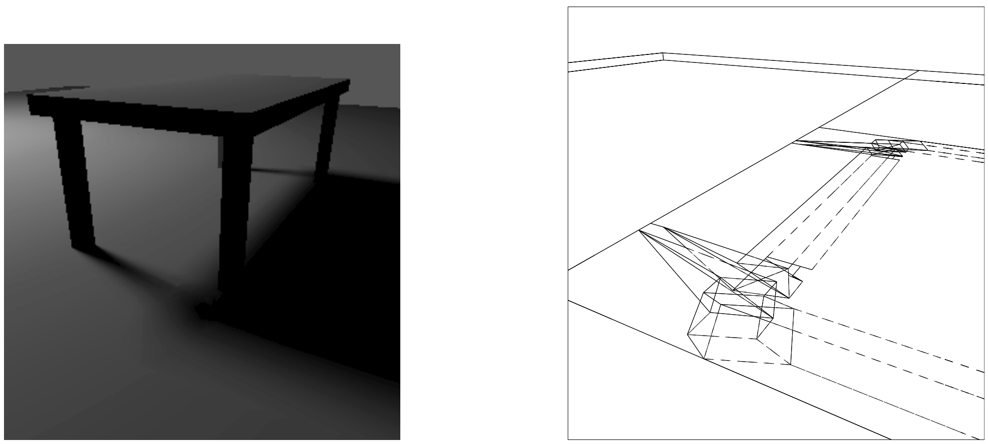


Figure 5-17. Image and Mesh Degree

The function `select_degree_on_edge( e )` applies the criteria described above for edges of the mesh, while `select_degree_on_interior_edge( te )` applies the criteria described for edges interior to mesh faces. The choice of basis functions and their construction is described in the following section.

5.4.2 Basis Function Design

The basis functions chosen to interpolate the radiance are triangular Lagrange polynomials. For triangles in which all three edges are linear, the method of plates is used to construct a linear basis function over the triangle (Prenter [44]). Assume the (two-dimensional) vertices of the triangle are numbered $\{p_1, p_2, p_3\} = \{(x_1, y_1), (x_2, y_2), (x_3 y_3)\}$. At each point p_i three points are used to create a plane that has equation $Q_i(x, y)$ that is the basis function $\phi_i(x, y)$. These triples are for the respective points:

$$
\begin{aligned}
p_1, &\quad ((x_1, y_1, 1),\ (x_2, y_2, 0),\ (x_3, y_3, 0)) \\
p_2, &\quad ((x_2, y_2, 1),\ (x_3, y_3, 0),\ (x_1, y_1, 0)) \\
p_3, &\quad ((x_3, y_3, 1),\ (x_1, y_1, 0),\ (x_2, y_2, 0))\ .
\end{aligned}
\tag{5-4}
$$

The interpolating function is defined as:

$$s_N(x, y) = \sum_{i=1}^{n} f(p_i)\, \phi_i(x, y)\ , \tag{5-5}$$

where $f(p_i)$ is the radiance value at point p_i. For the linear interpolant, n is equal to 3. For this interpolant the formal error bound is given by (Prenter [44]):

$$max|\ f(p) - s_N(p)\ | \le 4M_2h^2, \tag{5-6}$$

where h and M_2 are the same as in Eq. (5-3). The triangular linear interpolant, as well as each linear basis function is an equation of the form:

$$s_N(x, y) = a_i x + b_i y + c_i \tag{5-7}$$

For triangles on which all edges are of degree 2, a six-point interpolant is used (Prenter [44]). At each vertex a quadratic equation of the form:

$$P_i(x, y) = a_i x^2 + b_i xy + c_i y^2 + d_i x + e_i y + f_i, \tag{5-8}$$

is used as a basis function. To create these, at each vertex p_i two planes $Q_i(x, y)$ and $\tilde{Q}_i(x, y)$ are created, using the method of plates described above. The basis function for vertex p_i is defined by the multiplication of the two planes:

$$\phi_i(x, y) = Q_i(x, y)\, \tilde{Q}_i(x, y)\ . \tag{5-9}$$

The construction is illustrated in Figure 5-18(a).

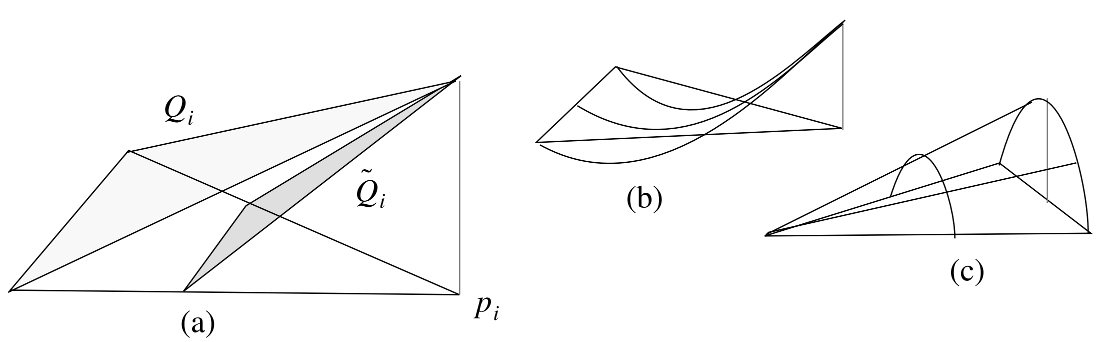


Figure 5-18. Basis Function Construction

Such basis functions are defined at the three vertices of the triangle, and at one interior point on each of the edges. The resulting interpolant is again given from Eq. (5-5), where n is equal to 6. Two different types of basis function result, which are illustrated in Figure 5-18(b) and (c). Those defined at a vertex are called quadratic vertex basis functions (Figure 5-18(b)), while those at the interior points are called quadratic interior point basis functions (Figure 5-18(c)).

A slightly more complicated situation arises when a triangle has 1 or 2 linear edges, and 2 or 1 quadratic edges respectively. Since it is necessary to maintain C^0continuity, appropriate basis functions must be designed to satisfy this requirement.

The required basis functions must be such that the interpolant is linear along the linear edges, yet quadratic along the edges of degree 2. Consider for example the vertex in Figure 5-19(a), in which the right hand neighbouring edge is linear, while the left hand edge is quadratic. The desired basis function should have the form shown in Figure 5-19(b). Two achieve this, the basis function is constructed by calculating the coefficients for Eq. (5-8), using the six-point algorithm, on the following six three-dimensional points:

$$(x_1, y_1, 1), (x_2, y_2, 0.5), (x_3, y_3, 0), (x_4, y_4, 0), (x_5, y_5, 0), (x_6, y_6, 0) \tag{5-10}$$

Notice that forcing the basis function to interpolate the value $\{1, 0.5, 0\}$ along the edge $\{(x_1, y_1), (x_3, y_3)\}$ effects linear interpolation on that edge. The basis function defined from the six-point quadratic interpolant of the point in Eq. (5-10), is called a linear/quadratic basis function.

For triangles requiring one linear edge, two linear/quadratic basis function are used at the vertices of the linear edge, while the remaining vertex is assigned a quadratic vertex basis function, and two quadratic interior point basis functions along the quadratic edges. A total of 5 function values are required for the triangle with one linear edge.

For triangles with two linear edges, all vertices have linear/quadratic basis functions, and an additional quadratic interior point basis function is defined at the point along the quadratic edge. A total of 4 function values are required for this interpolant.

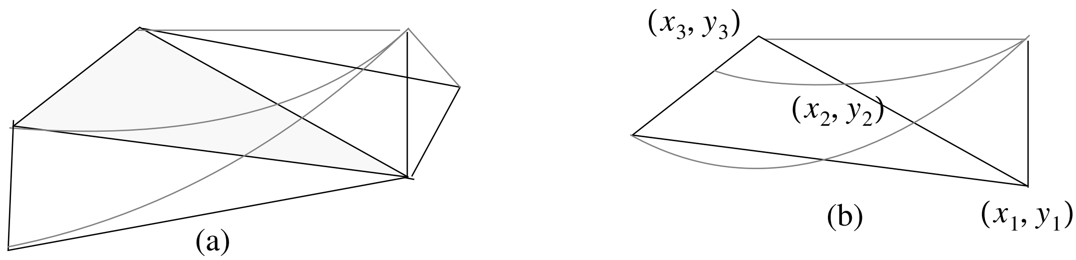


Figure 5-19. Linear/Quadratic Basis Functions

5.4.3 An Interpolant for Triangles at a Singularity

A typical singular vertex will have many edges joining at that point, as described in Section 5.3.5 The value at the vertex is defined as a limit of the function as it approaches the singular point, and therefore it is multi-valued. To represent this using interpolants is somewhat difficult. To deal with the problem, Tampieri [60] suggested that degenerate tensor product interpolants be used to represent the radiance in such triangles.

A similar approach is introduced here, but instead of a degenerate tensor product, an inverse bilinear mapping is used to map points from the triangular domain to a canonical tensor product defined on the $\{ [0, 1] \times [0, 1] \}$ domain. This allows the representation of the different radiance values along the edges joining at the vertex.

The construction starts by determining the number of edges joining at the singular point. The total angle θ_{total} of the incoming triangles is computed (see Figure 5-20(a)). The value of the unoccluded illumination at the singular vertex L_{max} is also computed.

For triangle i, whose edges joining at the singularity form angle θ_i, the values of radiance for the biquadratic tensor products along the edge corresponding to the singular point are assumed to vary in the interval:

$$\left[L_{max}\left(\sum_{j=0}^{i-1} \theta_j / \theta_{tot}\right),\ L_{max}\left(\sum_{j=0}^{i} \theta_j / \theta_{tot}\right) \right]. \tag{5-11}$$

This is an approximation, since it would be necessary to compute the limit values of radiance along each edge to determine the exact range of radiance in each triangle. The midpoint vertex in the tensor product is assigned the average of these two values. In this fashion a C^0 continuous representation is constructed for the radiance around a singularity. The construction is illustrated in Figure 5-20.

The inverse bilinear transformation is single valued, allowing points (x, y) to be mapped to canonical point in the tensor product domain (s, t):

$$s(x, y) = (c_1 x + c_2 y + c_3) / (c_4 x + c_5 y + c_6), \quad t(x, y) = (c_7 x + c_8 y + c_9) / c_{10} \tag{5-12}$$

We derive the constants in Eq. (5-12) by solving the bilinear map for (s, t).

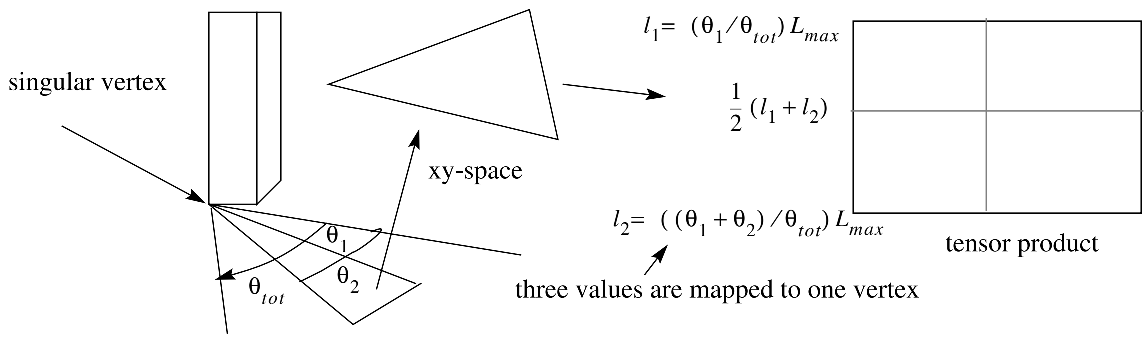


Figure 5-20. Interpolant Construction at a Singularity

5.4.4 Error Bounds for the Triangular Interpolants

The theoretical error bounds for the linear and quadratic triangular interpolants have been given in Eqs. (5-2) and (5-6). The problem with these error bounds, as mentioned above, is their dependence on the maximum absolute value of the derivatives within the domain of interest. In some faces of the discontinuity mesh the derivatives are sufficiently small to permit these bounds to be meaningful. Identifying these cases however is expensive, and not a practical way of assessing quality.

For monotonic faces, a simple error bound is given from the maximum difference between radiance values at the three points of the polygon. The error bound B of the interpolant s_N, on triangles over which radiance is monotonic, is given as:

$$B = max| f(p_i) - f(p_j) |, \quad i, j = 1, 2, 3. \tag{5-13}$$

For edges for which adaptive subdivision has been performed, the error bound of the approximation along each edge is given by the analysis of the adaptive subdivision method in

Section 3.4.2.2. The error bounds depend on the degree and position of the interpolants and are given by Eqs. (3-8), (3-9) and (3-10).

Error bounds have not been strictly established for non-monotonic faces for which adaptive subdivision has not been performed. However, for non-monotonic faces with a single extremum, the quality of the interpolant can be estimated closely by the maximum of B in Eq. (5-13) and the maximum difference of the radiance values of the interior points along each edge of the triangle used in the quadratic interpolant construction. As was noted in the statistics, the non-monotonic faces represent only a small percentage of the total number. In addition, they are usually small faces and thus tend not to cause significant error in the approximation.

5.5 The Combination of Penumbral and Unoccluded Regions

The structured algorithm of Chapter 3 can be used in regions of a receiver which are completely unoccluded. The algorithm as introduced for the unoccluded case builds a grid of tensor product interpolants over a region that is in light. We now wish to use the construction for the portions of the environment that are unoccluded in combination with the interpolant representation for illumination in the penumbra previously presented.

Using the simpler tensor product representation in the unoccluded regions allows the representation of illumination to be compact, since these portions need not be triangulated. The careful error bound analysis presented in Chapter 3 is applied when building these interpolants, which consequently satisfy tight and relevant error bounds.

The interpolant of degree n defined on a triangle has zero mixed $(n+1)^{\mathrm{st}}$ derivatives. On the other hand, tensor product interpolating formulations do not have this problem. This property can be important when the interpolating representation is used in the global solution.

5.5.1 Combining Partially Occluded and Unoccluded Structured Interpolants

The combination of occluded and unoccluded interpolants is performed in two stages. First, the algorithm segments the environment into regular light regions. These are processed using the same algorithm as for the unoccluded case. The second step involves segmenting remaining, irregular regions of light.

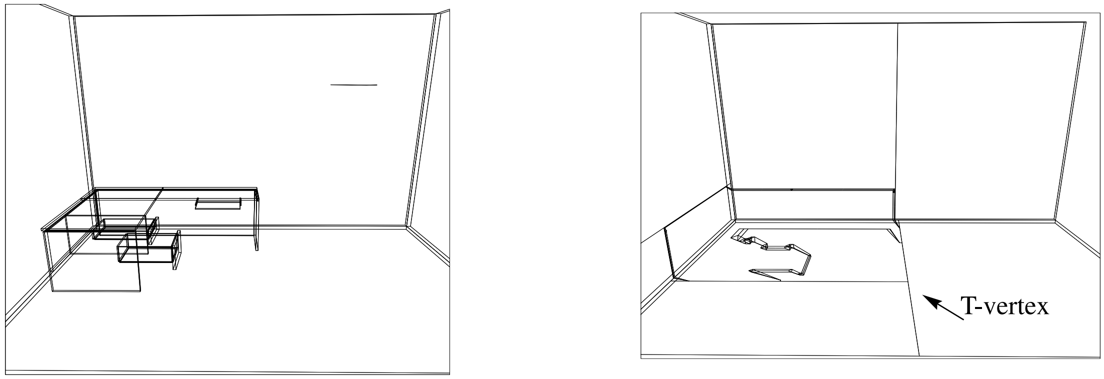
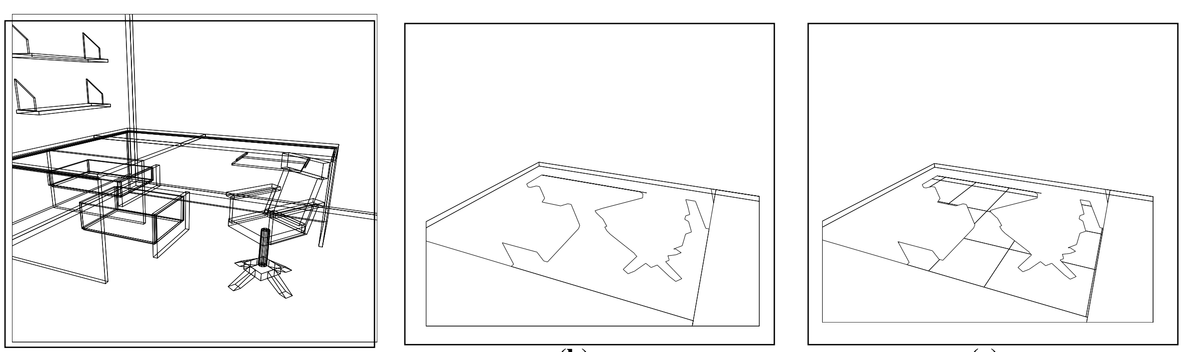


Figure 5-21. Segmenting the Penumbra into Light Regions

Regular Light Regions

To allow the combination of the occluded and unoccluded approaches, the bounding box of each penumbral group is first calculated. The resulting light regions can be segmented in a regular fashion based on the collection of bounding boxes (see for example Figure 5-21(b)).

For each original polygon of the scene, if its associated mesh contains an unoccluded region, the maximum and inflection point analysis is performed once, and the lists `{list_x,list_y}` constructed with the methods of Section 3.4, are stored with the surface. For each rectangular region created from the bounding box segmentation, the lists are reparameterised into the smaller rectangular region, and appropriate tensor product interpolants are built. Thus the expense of calculating the maximum and the inflection points is incurred only once, and only for the cases in which it is required.



(a) **(b)** **(c)**

Figure 5-22. Irregular Light Regions

Irregular Light Regions

Despite the segmentation into regular light regions using the bounding boxes of each penumbral group, there will almost always be certain unoccluded regions that have irregular, and possibly concave, shapes. A typical example of this is shown in Figure 5-22(b), for the scene shown in Figure 5-22(a). A straightforward triangulation of such a region can result in many small and ill-shaped triangles. Such triangulations present numerical problems, and problems of rendering efficiency. In addition the benefit of the error bounds applicable to the regions with tensor product interpolants cannot be realised.

The structured algorithm used for the unoccluded case can however be used again to segment these (possibly large) irregular regions. The lists `{list_x,list_y}` defined by the adaptive subdivision for unoccluded illumination are used to segment the irregular regions into better shaped regions (see Figure 5-22(c)). In many cases, these regions can become regular rectangles, for which the tensor product interpolant method is used. For the remaining regions, which may still be concave, the triangulation approach used for the penumbra is used. However, since the segmentation is based on the error bound analysis for the unoccluded radiance, the radiance within these triangles can be expected to be well behaved.

When tensor product interpolants are built using this algorithm, they inherit the analysis performed for the simple unoccluded case. As a result, the error bounds given from Eqs. (3-7),(3-8), (3-9) and (3-10) apply to these interpolants, depending on which region of the unoccluded radiance function they correspond to.

5.5.2 Continuity for the Combined Interpolants

After the creation of the regular unoccluded regions and the segmentation of the rest, a complete partition of the scene polygon surfaces has been achieved. In every region of this partition an interpolant exists representing radiance from the diffuse source.

The peculiarity of this formulation is that parts of the domains are tensor products, and parts are triangular interplants. As a result, the interpolant boundaries will not always match, that is there may be T-vertices where triangular and tensor product domains meet. To avoid this, continuity must be guaranteed along these edges. When constructing the triangular interpolant on an edge neighbouring a tensor product, two actions are performed. First the degree of the edge of the triangular interpolant is forced to match that of the tensor product. Second, instead of using the actual values $f(p_i)$ of radiance for the coefficients in Eq. (5-5), the values of the tensor product interpolant are used. Since the linear or quadratic interpolants through two or three points respectively are unique (Prenter [44]), continuity is guaranteed.

A similar situation occurs when two of the regularised light regions form a T-vertex (see for example Figure 5-21(b)). Again, the coefficients at the vertices of tensor products that lie on the boundary are taken to be samples of the neighbouring interpolant instead of the actual radiance value.

5.6 A Complete Structured Sampling Algorithm

The structured sampling algorithms for unoccluded illumination, the construction of the complete discontinuity mesh and the backprojections, and finally the algorithms presented in this chapter for the regions of penumbra can be used together to represent radiance from a diffuse light source with low-degree piecewise smooth polynomials. The combination of all these methods results in the complete structured sampling algorithm.

The first step in the algorithm is the construction of the discontinuity mesh, followed by the computation of the backprojection in each face. For surfaces that contain faces in light, the unoccluded structured sampling algorithm is run, and the resulting grid from the adaptive subdivision is stored with the face. The segmentation into regular light faces and the splitting of irregular light regions follows. Adaptive subdivision for singularities and large faces is then performed if necessary. If desired, mesh reduction and degree selection is performed to the user-specified tolerance, and the interpolants are finally constructed on the resulting mesh. The algorithm is summarised in the procedure shown in Figure 5-23.

The analysis of error bounds for the penumbral regions in Section 5.4.4 and the use of the analysis for unoccluded illumination to form the interpolants in the regions of light, results in the construction of an overall high quality interpolant representation of radiance for general environments. In addition this interpolant can be computed at reasonable expense. Finally, with the application of mesh reduction and degree selection, the number and degree of the interpolants is small.

This construction allows high quality images to be rendered at low cost, as well as the use of the interpolant to generate a polygonal mesh for interactive viewing on a modern graphics workstation. The interpolant representation is also good initial estimate for a global transfer solution, outlined in Chapter 6.

```
complete_structured_sampling ( Scene s, Source src)
{
    compute_mesh( s, src );
    compute_backprojections( s, src );
    for each receiver polygon p in s do
         Mesh m = get_mesh( p );
         if m has faces in light then
              { list_x, list_y } = find_max_and_create_lists();
              for list in { list_x, list_y } do
                   find_inflection_points( list );
                   adaptive_subdivision( list );
              done
              store {list_x, list_y} with p;
         endif
         segment_light_faces( m, src, list_x, list_y );
         adaptive_subdivision( m, src );
         reduce_mesh( m, area_tol );
         foreach regular light face f of m do
               build_2d_interpolant( list_x, list_y, f);
         done
         construct_penumbral_interpolants( m, degree_tol );
    done
}
```

Figure 5-23. The Complete Structured Sampling Algorithm

5.7 Rendering

In Section 3.2.1, two ways of rendering a scene with a polynomial representation of illumination were presented. For general scenes including penumbra, the meshes are much more complicated, and several new issues must be addressed. The expense of ray-casting such images is such that it is often worthwhile to render using graphical hardware for medium quality images. To achieve this, a method is described that allows the rendering of Gouraud-shaded polygons in hardware to approximate quadratic interpolants within an acceptable tolerance.

5.7.1 Ray-Cast Rendering

To render a ray-cast image of a scene in which an interpolant representation of radiance is assigned to each surface, it is necessary to locate the appropriate interpolant at each visible point. This is a very expensive operation if performed at every pixel, since it would require the entire mesh of the visible object be searched, adding to the already high cost of the ray-casting process for visible surface determination.

To improve on this naive approach, the last visible point and the corresponding face in the mesh is stored with each object. Thus the next time a visible pixel is located on the same object, the mesh can be traversed along the line from the previous visible point to the current point. Due to the adjacency information stored in the mesh, and the inherent proximity between visible pixels, this search will be on average very short.

5.7.2 Interactive Polygon-Based Rendering

The graphics hardware on modern workstations allows the rendering of several hundreds of thousands of Gouraud-shaded polygons per second. Since the polygons that will be rendered from the penumbral regions are triangles, Gouraud shading provides a close approximation to linear interpolation, and is an approximation to an interpolant of the form of Eq. (5-7). For quadratic or mixed linear quadratic elements, further subdivision of the triangles is required to approximate the effect of quadratic interpolation.

Recall that to render the tensor product interpolants, a mesh of quadrilaterals is used. The values at the vertices of these polygons are determined by resampling the tensor product. It is necessary to maintain continuity between the resampled quadrilaterals and the triangles from the penumbral domains.

Continuity Between Triangles and Tensor Products

The subdivision level for the quadrilaterals generated in the regions represented by tensor products can be determined in the same manner as that described below for the approximation of quadratics by Gouraud shaded triangles. Once this has been determined, the intersections of these triangles along the edges of neighbouring triangulated faces are found and stored with those faces. When these faces are rendered, the triangles actually sent to the pipeline always share edges with the respective polygons from the tensor products. This is shown with the dashed lines in Figure 5-24(a).

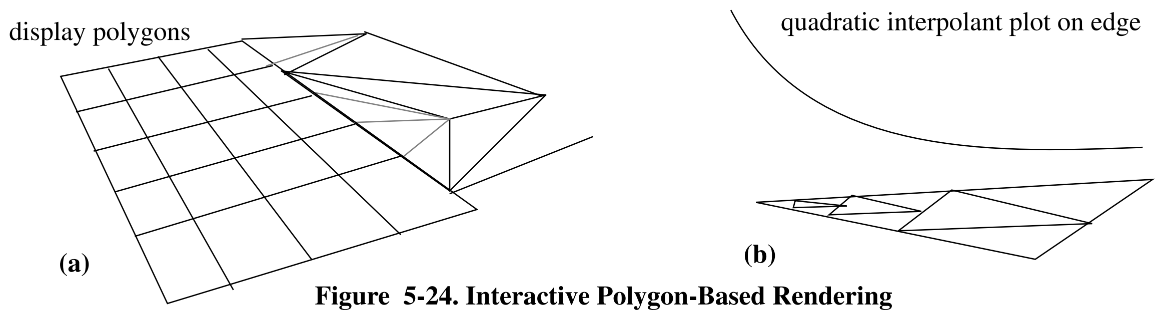


Figure 5-24. Interactive Polygon-Based Rendering

Approximating Quadratic Interpolants with Gouraud-Shaded Triangles

The evaluation of quadratic and linear functions can be performed very efficiently, therefore it is possible to perform satisfactory quality evaluation without incurring significant additional cost.

For each quadratic edge of a triangle to be rendered, call $s_2(t)$ the quadratic Lagrange polynomial interpolating radiance along that edge, and $s_1(t)$ the linear interpolant. The parameter t varies in $[0, 1]$. The maximum error along the edge is given as follows:

$$MaxErr = \max_{0 \le t \le 1} |s_2(t) - s_1(t)| \quad . \tag{5-14}$$

To find *MaxErr* it is sufficient to find the minimum value of $s_2(t)$ and then find the distance to the line $s_1(t)$. If this value results in a discrepancy larger than one pixel RGB value along any edge, the triangle is subdivided. The subdivision scheme used is to split a triangle into three subtriangles by splitting each edge at the midpoint (see Figure 5-24(b)).

5.8 Numerical Tests and Quality Evaluation

To evaluate the quality of the complete structured algorithm as well as the individual algorithms for mesh reduction and degree selection, a set of test suites is run, similar is spirit to the test suites run for the unoccluded structured sampling algorithm presented in detail in Section 3.7. The error metrics and methodology are the same as those used in the analysis of the unoccluded algorithm. The current implementation includes the combined structured algorithm with the treatment of singularities, mesh reduction and degree selection. Adaptive subdivision for polygon rendering is currently only performed around the singularity. Adaptive subdivision and unoccluded region regularisation have not yet been implemented.

Results are shown for the four test suites described in Section 5.2.3. Each suite consists of 5 tests, (Test 1-Test 5), involving a quadrilateral source illuminating a simple table with four legs. In the first sequence, one edge of the source grows (Grow sequence), in the second sequence the square source is lowered so that it comes closer to the blocker (Down sequence), in the third the square source is at the corner of the table (Corner sequence) and moves down. In the last sequence the square source moves away while maintaining the same height (Away sequence). The complicated scenes are a complex desk model in a room and the same model with the addition of a chair and shelves on the wall. The geometry of the Desk scene is shown in Figure 5-21(a) and for the Desk & Chair scene in Figure 5-22(a).

5.8.1 Results for Mesh Reduction and Degree Selection Algorithms

Three sets of tests are run. The first, reported in Table 5-4, shows the quality achieved by the structured algorithm without the use of mesh reduction or degree selection. Table 5-5 shows the statistics for the mesh reduction algorithm, while Table 5-6 shows the results for the degree selection approach.

The results for the simple algorithm (i.e., without mesh or degree reduction) are overall encouraging. For the “Grow” and “Away” sequences and for the complicated scenes the error is minimal. The larger errors observed in the “Down” and “Corner” sequences are due mostly to the reconstruction in the unoccluded regions rather than the penumbra. Some fine tuning is required for the tensor product reconstruction reparameterisation in the regular regions.

			Test 1		Test 2		Test 3		Test 4		Test 5	
Scene	**pixel error**	**RMS**	**pixel error**	**RMS**	**pixel error**	**RMS**	**pixel error**	**RMS**	**pixel error**	**RMS**	**pixel error**	**RMS**
Grow	1.77%	0.010	1.71%	0.010	1.96%	0.011	2.03%	0.010	1.35%	0.011	1.80%	0.007
Down	4.51%	0.013	9.83%	0.015	2.99%	0.010	5.53%	0.013	3.64%	0.019	0.58%	0.009
Corner	3.80%	0.020	1.42%	0.007	1.98%	0.009	3.84%	0.020	4.95%	0.026	6.82%	0.040
Away	1.45%	0.013	1.72%	0.008	1.67%	0.012	1.75%	0.015	1.91%	0.020	0.21%	0.011
Desk	0.04%	0.002	-	-	-	-	-	-	-	-	-	-
Desk & Chair	0.08%	0.003	-	-	-	-	-	-	-	-	-	-

Table 5-4. Tests for Simple Penumbra Interpolation

	Area Tolerance.01%of Maximum				Area Tolerance.08%of Maximum			
Scene	**pixel error**	**RMS**	**% pixerr Increase**	**% Faces of Orig.**	**pixel error**	**RMS**	**% pixerr Increase**	**% Faces of Orig.**
Grow (avg.)	1.82%	0.010	2.82%	53.63%	1.85%	0.010	4.52%	51.69%
Down (avg.)	4.26%	0.013	0.00%	59.13%	4.26%	0.013	0.00%	55.52%
Corner (avg.)	4.20%	0.021	4.22%	59.72%	4.17%	0.021	3.47%	59.08%
Away (avg.)	1.62%	0.022	11.72%	58.52%	1.45%	0.013	0.00%	55.59%
Desk	0.04%	0.002	0.00%	76.86%	0.06%	0.002	50.00%	76.23%

Table 5-5. Tests for Mesh Reduction Algorithm

The results for the mesh reduction algorithm in Table 5-5, indicate that the mesh can be reduced significantly (up to 48%), with only marginal reduction in quality. The mesh reduction achieved as well as the specific error incurred vary however from configuration to configuration. The “Grow” and “Down” sequences are more prone to the higher error, but again this is mainly due to errors in the reconstruction by the tensor products in the light regions.

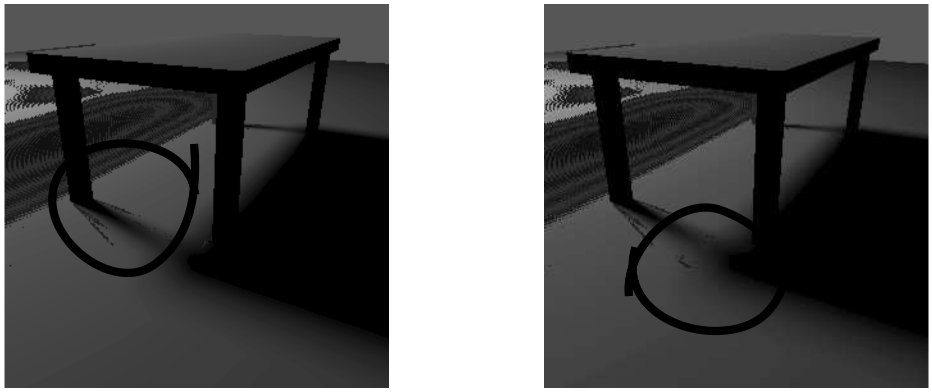
The results of the tests for the degree selection approach are shown in Table 5-6. The error can increase when the degree is reduced, but still at levels that are overall acceptable (less that 7% of the pixels have significant error). Thus error levels for the “Grow” and “Down” sequences are still in the acceptable 4-6% range, even though between 49% and 69% of the edges have linear degree.

Scene	Linearisation Tolerance 5 RGB Levels				Linearisation Tolerance 15 RGB Levels			
	% pixerr	RMS	% pixerr Increase	% Linear	% pixerr	RMS	% pixerr Increase	% Linear
Grow (avg.)	2.46%	0.014	38.98%	28.19%	6.59%	0.016	272.32%	63.23%
Down (avg.)	4.97%	0.013	10.20%	36.28%	5.77%	0.014	27.94%	69.51%
Corner (avg.)	3.89%	0.020	2.37%	23.74%	4.49%	0.021	18.16%	49.09%
Away (avg.)	1.79%	0.014	23.45%	49.95%	2.81%	0.016	93.79%	79.62%
Desk	0.23%	0.010	21.05%	13.95%	0.59%	0.010	210.53%	33.91%
Desk & Chair	0.45%	0.013	7.14%	17.66%	0.89%	0.015	111.90%	39.64%

Table 5-6. Tests for Degree Selection Algorithm

5.8.2 Images Generated Using Structured Sampling in the Penumbra

Figure 5-25(a) is the scene of Figure 5-13, for which the error has been superimposed as dark splotches. Apart from the error in the lit region, the only visible error in the penumbra appears close to the singularity, which is marked with the thick line. Adaptive subdivision is expected to solve this problem. Similarly Figure 5-25(b) is Figure 5-14 with the error superimposed. Despite mesh reduction of 46%, little additional error appears (again marked with a thick line). The degree selection algorithm, which has classified 50.7% of the edges as linear, does display a little more error. This can be seen in Figure 5-26, which is the error image for Figure 5-17. Additional error appears close to the singularity and in regions close to light. Recall that Figures 5-13, 5-14 and 5-17 appear in Appendix C as Colour Images 7(a), 8(a) and 9(a) respectively. Also Figures 5-25(a), (b) and 5-26 appear as Colour Images 7(b), 8(b) and 9(b) respectively. The error is marked in red and can be seen much more clearly in colour.



(a) (b)

Figure 5-25. Error Images (a) for Simple Interpolant (b) for Reduced Mesh

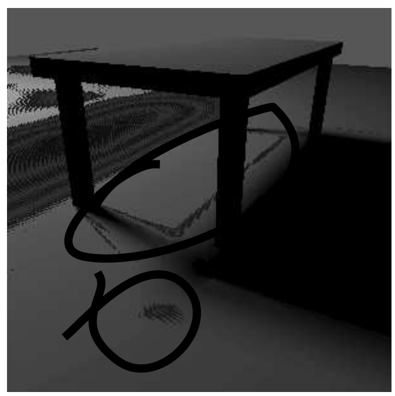
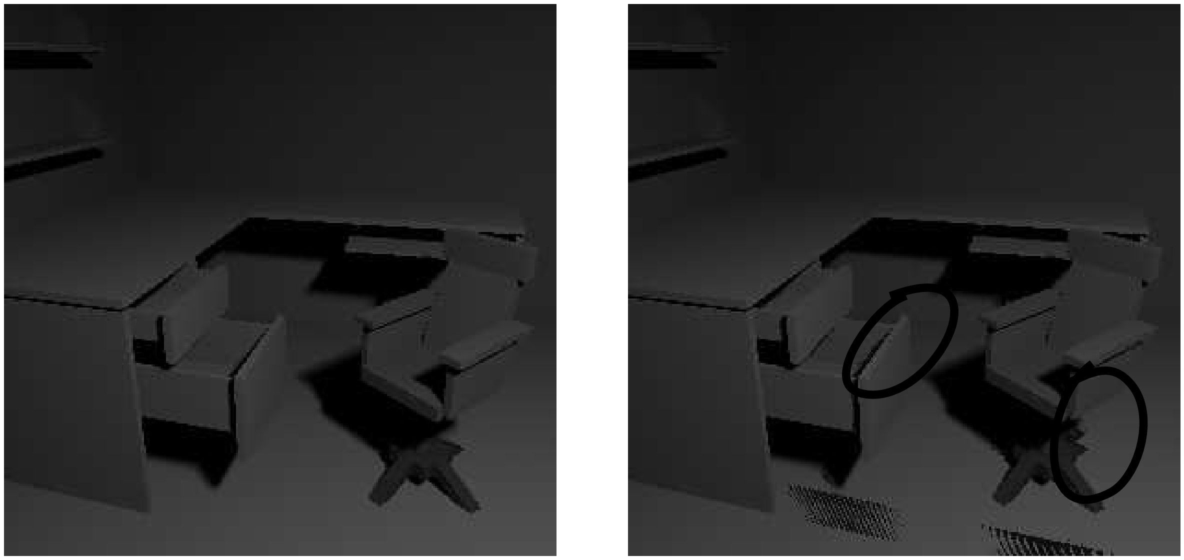


Figure 5-26. Error Image after Degree Selection

A more complex scene incorporating a desk and chair is shown in Figure 5-27(a), with the error superimposed in Figure 5-27(b), which correspond to Colour Images 10(a) and (b). In the colour images the error is marked in red and is more visible. Again, some error is visible in the unoccluded areas, due to the triangulation of the irregular regions. With the application of the region regularisation, this problem will be resolved. Some error is visible close to the singular vertices of the chair base. Some error appears in the penumbra of the drawers.



(a) (b)

Figure 5-27. Desk Scene and Error

5.8.3 Discussion of Numerical and Visual Results

The numerical and visual results presented are very encouraging. The combined structured algorithm performs very well in its current form, and with the addition of adaptive subdivision and some fine tuning for the tensor product reparameterisation, even better results are expected.

The mesh reduction algorithm has presented remarkably good results over a significant number of scenes. For high mesh reduction, the increase in error is in general minimal, indicating that the size of the complete discontinuity mesh is much more than required for satisfactory reconstruction. However, there are cases in which the reconstruction quality can degrade, particularly when T-vertices are left in the mesh which force the creation of small or elongated triangles. Further geometric manipulation of the mesh can be used to eliminate these artifacts. One issue that is more difficult to address is that of animation. No provision is made for consistency in mesh reduction, and thus over animated sequences flickering can occur as the mesh changes from frame to frame. This can be addressed in the context of a more general incremental meshing algorithm.

For degree selection, the results are also very encouraging. Numerical and quantitative criteria are good indicators of the quality achieved by linear interpolation. Nonetheless, the method can be improved by incorporating some criterion based on the possible visible impact of the degree reduction, to avoid occasionally objectionable artifacts.

5.8.4 Comparison of Ray-Casting and the Interactive Polygon Rendering Algorithm

As mentioned above the current implementation of polygon rendering only performs adaptive subdivision close to the singularity. Given this limitation, the same set of image tests was rendered using the polygon rendering approach and the ray-cast approach to determine the quality achieved by polygon rendering algorithm. Figure 5-28 shows a frame from the Grow sequence rendered by the polygonal method Figure 5-28 (a) and the ray-casting method Figure 5-28(b). Little difference can be seen between the two images. This is also true when these images are viewed on a CRT display.

Statistics showing the pixel and RMS error for the polygonal rendering are shown in Table 5-4. When compared to the error shown by the ray-cast images in Table 5-4, it can be seen that the polygonal rendering is worse in qualitative terms. Despite this, the visual result is often satisfactory. It must be stressed that the average ray-cast image will take a few minutes to render, while the polygon rendering hardware can render an image in close to real time (a few frames a second).

(a) (b)

Figure 5-28. Polygon Rendering

			Test 1		**Test2**		**Test 3**		**Test 4**		**Test 5**	
Scene	**pixel error**	**RMS**	**pixel error**	**RMS**	**pixel error**	**RMS**	**pixel error**	**RMS**	**pixel error**	**RMS**	**pixel error**	**RMS**
Grow	3.93%	0.038	3.28%	0.049	4.16%	0.048	4.70%	0.048	7.51%	0.048	0.00%	0.000
Down	5.74%	0.054	11.06%	0.047	3.30%	0.049	5.81%	0.053	4.86%	0.059	3.68%	0.063
Corner	7.52%	0.101	5.06%	0.082	6.26%	0.089	7.92%	0.099	8.64%	0.109	9.74%	0.126
Away	2.62%	0.046	3.14%	0.040	2.98%	0.042	2.81%	0.045	3.35%	0.050	0.81%	0.051
Desk	1.67%	0.010	-	-	-	-	-	-	-	-	-	-
Desk & Chair	1.23%	0.013	-	-	-	-	-	-	-	-	-	-

Table 5-7. Error of Polygon Rendering

5.9 Summary and Discussion

In this chapter, the full structured sampling algorithm was presented. The algorithms developed for unoccluded illumination in Chapter 3 were combined with the mesh and backprojection construction of Chapter 4 to create an accurate and efficient representation of radiance from a diffuse light source in environments with occlusion.

An extensive empirical and numerical study of radiance in penumbral regions was performed. It was found that in the majority of cases, radiance in the penumbra is monotonic and amenable to reconstruction by linear or quadratic interpolants. Configurations that cause the appearance of extrema or irregular behaviour were characterised, and methods presented to identify such behaviour efficiently.

The understanding of radiance function structure within penumbral regions was used to guide the construction of an interpolating, piecewise polynomial representation of the function. The task of building the interpolant was segmented into two main components: that of constructing appropriate domains for the interpolants, and that of choosing and designing appropriate basis functions.

For construction of the interpolant domains, the complete discontinuity mesh was used as a starting point. The faces of the mesh are triangulated, and the radiance information is stored compactly with the mesh. The observations made from the empirical study suggested that many of the edges in the mesh are not actually required for satisfactory reconstruction of illumination in the penumbra. To benefit from this, a mesh reduction algorithm was introduced, based on the removal of faces with small area and faces that cause concave regions in the mesh. The more infrequent cases for which adaptive subdivision is required were identified, and an adaptive subdivision algorithm presented, that uses the tight error-bound driven algorithms developed for the unoccluded case.

From the same empirical observations, it was determined that in many cases linear interpolation is sufficient for illumination reconstruction in the penumbra. An algorithm was presented which characterizes the edges in the mesh and the triangulation as requiring linear or quadratic degree polynomials to achieve high quality reconstruction. This selection is based on properties of the corresponding backprojections of the faces adjacent to the edges under consideration and the position of each edge in the mesh. A set of basis functions was designed that allow the use of mixed degree polynomials for reconstruction. Error bounds for the triangular interpolants were also discussed.

The triangular interpolant algorithms were then combined with the tensor product formulation for the unoccluded case to create the interpolant for radiance across the whole scene. The resulting combination completes the structured sampling algorithm using the results for unoccluded illumination, the mesh and backprojection algorithm and the algorithm for the penumbrae. Rendering of scenes for which the interpolant has been built is then discussed.

Numerical tests were performed for a combination of simple environments, for which geometric parameters such as distance and angles between objects were modified, and certain more complex scenes, for which more complicated shadow structures appear. The results show that both the mesh reduction and the degree selection algorithm can be applied without significant degradation of quality in the reconstruction. Also the complete structured sampling algorithm results in high quality reconstruction of radiance for images in scenes with occlusion.

Chapter 6

A Structure-Driven Global Illumination Algorithm

The structured sampling algorithms presented in the previous chapters result in the construction of a piecewise polynomial representation of radiance due to direct illumination from a light source. In this chapter we show that this representation is suitable for the solution of the global illumination problem using a higher-order finite element approach, specifically the Galerkin method. An iterative solution process is proposed similar to the progressive refinement algorithm in the constant radiosity-based methods.

Before presenting the structured global illumination formulation, we first show how meshes from multiple light sources can be merged. The merged mesh is subsequently used in the formulation of the global solution. The Galerkin higher-order finite element method is reviewed, as well as iterative relaxation solution methods. We then cast the global illumination problem in terms of these methods, using the mesh created by the structured sampling algorithms.

We also show that the structure-driven global illumination algorithm has several advantages over previous higher order methods. The interpolant representation of radiance due to direct illumination created by the structured algorithm can be used as an accurate initial approximation for an iterative solution. In addition, an extension of the discontinuity meshing algorithm facilitates accurate and efficient computation of the integrals required for the iterative solution. Finally the sample placement algorithms, developed in previous chapters can be used to adaptively modify the set of nodes over which the solution is calculated, effecting more efficient and accurate solutions.

6.1 Merging Meshes from Multiple Sources

The structured sampling algorithms previously developed can be applied directly to achieve merging of meshes from multiple sources. The merging process is a three step approach that requires a few minor modifications to the algorithms and data structures previously used. It is assumed that the mesh for the first source σ_1 (which is the source with the greatest power) was created using the structured method of the previous chapters. For the subsequent sources $\{\sigma_2, \ldots, \sigma_{i-1}\}$ the steps described below are followed. The procedure described next shows how the merged mesh for source σ_i is constructed.

Step 1: Computing the Mesh for σ_i

The complete structured sampling algorithm is applied for source σ_i, but the resulting mesh (tensor product and triangular portions), is stored in a separate data structure associated with each surface in the environment. The mesh created by previously treated sources is denoted by M_p and the new mesh, corresponding only to σ_i, is M_i.

Step 2: Merging the Meshes

In the second step, the two meshes M_i and M_p are merged, using the following set of rules.

1. If a tensor product region of M_i falls entirely within a tensor product region of M_p, the original region of M_p is subdivided into a set of finer tensor products, following the subdivision lines of the tensor product of M_i. The merged mesh has two sets of nodes, those originally belonging to M_p and those originally belonging to M_i. For each node, the other interpolant is sampled at that point, and the coefficients are added. If ringing is observed, further subdivision is performed. This construction is shown in Figure 6-1(a).

2. If a tensor product region of M_i falls within a penumbral region or overlaps a penumbral region of M_p, or vice versa, the domains are merged. A special flag is assigned to the edges that constituted the original tensor product domain, that is subsequently used to simplify the mesh (see Step 3). This construction is shown in Figure 6-1(b).

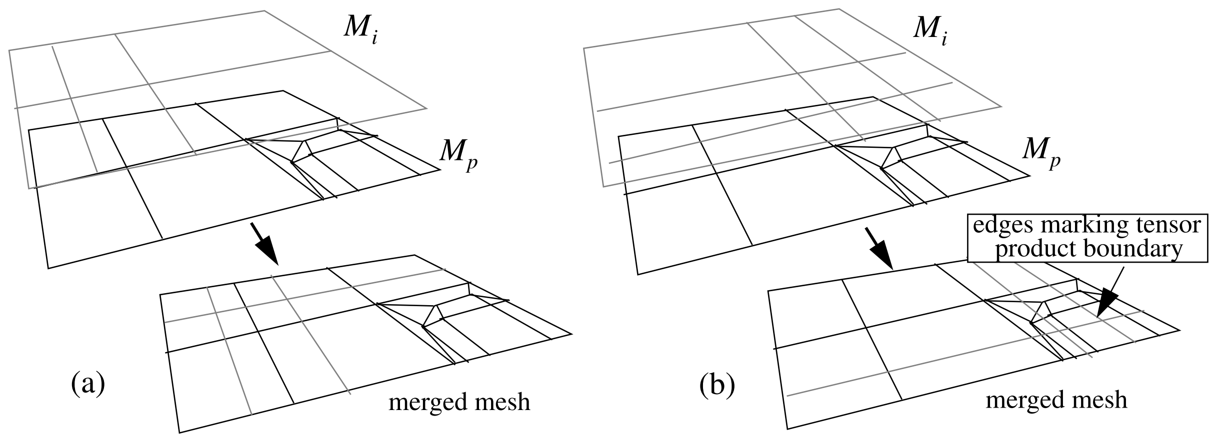


Figure 6-1. Mesh Merging Cases

3. If two penumbral domains overlap, the meshes are merged by adding the faces of M_i into the mesh of M_p.

All the merging operations can be performed efficiently since the adjacency information of each mesh allows all insertions to be performed without an expensive point location search, other than the location of the first point for each mesh merge.

Step 3: Simplifying the Resulting Mesh

The merged mesh will in general be somewhat more complicated than the previous mesh M_p. A variation of the mesh reduction algorithm presented in Section 5.3.4 is used to simplify the merged mesh.

All the criteria used in the original mesh reduction algorithm are applied to the merged mesh. In addition to the pass that removes concave faces, a pass is made in which it is attempted to restore as many tensor product domains as possible. This is done by finding the regions surrounded by the specially marked edges in accordance to rule 2 of Step 2, and by examining the relative behaviour of radiance in the penumbral region, and in the original tensor product. If the function can be well represented by the tensor product alone, the faces of the penumbral domain are removed and corresponding simpler tensor product interpolants formed.

6.2 Preliminaries for Higher Order Solution Methods

To comprehend the solution techniques used, it is necessary to provide some of the intuition behind the higher order finite element methods, and how these methods relate to the construction performed in the previous chapters. A brief introduction to the Galerkin method and the Southwell relaxation iterative solution algorithm for linear systems is also presented, since they form the basis for the structured solution to the global illumination problem. The previous higher order and discontinuity meshing approaches are then briefly reviewed.

6.2.1 Basis Functions and Function Spaces

The notion of a basis function can be viewed in different ways, depending on the application. In the previous chapters, a set of element "basis functions" was built over a single *element*, i.e., a triangle or a quadrilateral in the mesh constructed by the structured sampling algorithm. For the purposes of a finite element formulation however, a different approach is required.

For each surface in the environment a mesh exists, consisting of triangles and quadrilaterals. Each vertex or node in that mesh is adjacent to a set of elements over which interpolants are built, in the context described above and in Chapters 3 and 5. Let P the set of all such nodes be

$$P = \{p_1, p_2, \ldots, p_n\} . \tag{6-1}$$

Of the element "basis functions" defined in the previous manner on each element, a number of them will meet at each of these nodes. To see this consider the example of a linear triangular domain shown in Figure 6-2. In the context of a finite element formulation, each of the element "basis functions" meeting at a node is considered as a single basis function, with index *i*, corresponding to node p_i.

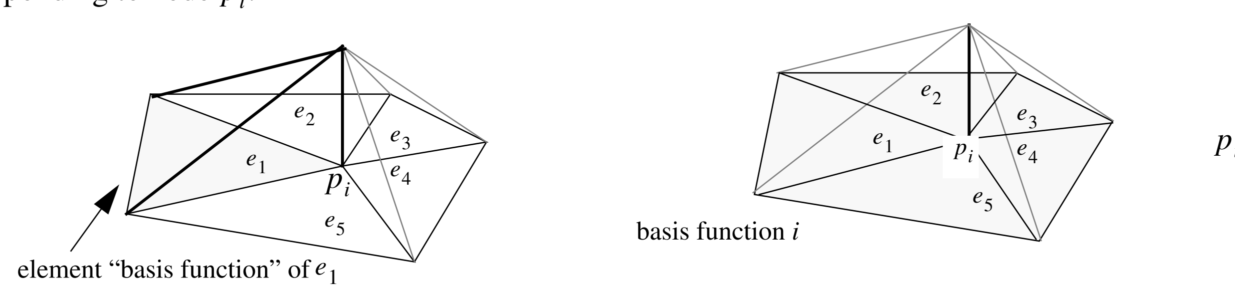


Figure 6-2. Different Basis Function Indexing

Each such basis function h_i is defined over the union of elements around each node, in a piece-wise fashion. The value of $h_i(s, t)$ depends on which element the point (s, t) is found. The region $\Omega_i = \bigcup_{i=1}^{5} e_i$ is called the *domain* of the basis function h_i. This construction gives a unique global indexing of each node, and consequently of the basis function $h_i(s, t)$, as well as the coefficient b_i corresponding to that node, used in the construction of the approximation. We assume that there are *n* such basis functions, spanning the entire domain of surfaces.

Let $\tilde{B}_e(s, t)$ be the interpolant over element *e*. Using the notation of previous chapters, the value of radiance at a point (s, t) is given by the expression

$$\tilde{B}_e(s, t) = \sum_{k=1}^{K_e} b_e^k h_e^k(s, t), \tag{6-2}$$

assuming there are K_e element "basis functions" defined over element *e*. Expression (6-2) is the same as Eq. (5-5) in Section 5.4.2. Using the global indexing scheme, there will be a set of global indices $S_e = \{i_1, i_2, \ldots, i_{n_e}\}$, corresponding to basis functions that are non-zero over the element *e*. From now on, the global indexing and corresponding basis functions will be used, unless explicitly stated elsewhere.

Using the global indexing expression, Eq. (6-2) becomes:

$$\tilde{B}_e(s, t) = \sum_{i=1}^{n} b_i h_i(s, t) = \sum_{i \in S_e} b_i h_i(s, t), \tag{6-3}$$

assuming there are *n* basis functions.

Function Spaces

Consider the set of functions $\{x(t)\}$ defined over an interval (a, b) such that:

$$\int_a^b |x(t)|^2 dt < \infty. \tag{6-4}$$

This set is the $L^2(a, b)$ function space and the functions $x(t)$ are L^2 functions. The functions of radiance that we are dealing with are L^2 functions. The inner product operator $\langle x, y\rangle$ defined over the function space, is:

$$\langle x, y\rangle = \int_a^b x(t)\, y(t)\, dt. \tag{6-5}$$

Two functions are orthogonal with respect to an inner product if

$$\langle x, y \rangle = 0. \qquad (6\text{-}6)$$

Finally the L_2 norm is defined as follows:

$$\|x\|_2 = \left(\int_a^b |x(t)|^2 dt\right)^{1/2} = \langle x, x \rangle^{1/2}. \qquad (6\text{-}7)$$

These definitions will be used in the subsequent formulations.

6.2.2 The Galerkin Method

The following presentation is adapted from Delves and Mohammed ([18]) to clarify the Galerkin approach and allow the presentation of the new algorithm within this framework. The presentation is given for one-dimensional functions, but the generalisation to the two-dimensional functions is straightforward. Consider the following integral equation, where the function $x(s)$ is unknown, and $x, y \in L^2(a, b)$:

$$x(s) = y(s) + \int_a^b K(s, t)\, x(t)\, dt \qquad (6\text{-}8)$$

This equation is a Fredholm integral equation of the second kind. The integration operation can be viewed as a linear operator, Ox on the space of functions:

$$Ox = \int_a^b K(s, t)\, x(t)\, dt. \qquad (6\text{-}9)$$

We can now rewrite Eq. (6-8) using the operator notation:

$$x(s) - \int_a^b K(s, t)\, x(t)\, dt = y(s) \Leftrightarrow (I - O)\, x = y. \qquad (6\text{-}10)$$

Substitute the operator $(I - O)$ with the operator L, and the original equation to be solved, can be written in the more compact format:

$$Lx = y. \qquad (6\text{-}11)$$

In general (6-11) has no analytic solution. To approximate a solution, the following related equation is solved:

$$x_N(s) = y(s) + \int_a^b K(s, t)\, x_N(t)\, dt, \qquad (6\text{-}12)$$

where the function $x_N(s)$ is an approximation to the true solution. The error function of this solution is defined as

$$\varepsilon_N(s) = x(s) - x_N(s). \qquad (6\text{-}13)$$

The function $x_N(s)$ is usually a simpler function, such as a piecewise polynomial. The residual of the approximation is defined:

$$r_N(s) = y(s) - Lx_N(s) . \tag{6-14}$$

It can be shown that (see Delves and Mohammed [18]), where $\|\cdot\|$ is the L_2 norm,

$$\|\varepsilon_N\| \le \frac{\|r_N\|}{1 - \|K\|}, \tag{6-15}$$

if $\|K\| < 1$. Thus minimising the residual provides a good approach to solving the integral equation. Residuals are also used extensively in the solution of linear systems of equations, which we will be returning to later. Consider the approximating function

$$x_N(s) = \sum_{i=1}^{n} a_i h_i(s) . \tag{6-16}$$

The functions $h_i(s)$ are assumed to be a set of basis functions for the function space in which we are searching solutions to Eq. (6-12). The Galerkin solution method requires that

$$r(s) = 0. \tag{6-17}$$

For an infinite basis this is equivalent to requiring that the residual $r(s)$ be orthogonal, with respect to the inner product, to each basis function $h_i(s)$, $i = 1, \ldots, \infty$. For a finite set containing n basis functions, a sufficient condition is that the residual be orthogonal to each basis function. This requirement gives rise to a system of n equations:

$$\langle h_i, r\rangle = 0 \quad \Leftrightarrow \int_a^b h_i(s)\,[y(s) - Lx_N(s)]\,ds = 0, \quad i = 1, \ldots, n \tag{6-18}$$

Eq. (6-18) can be written in matrix form:

$$\mathbf{L}_G \mathbf{a} = \mathbf{y}_G. \tag{6-19}$$

The quantity $\mathbf{a}$ is the vector of coefficients of the n basis functions. This is the vector of unknowns in the system to be solved.

The matrix $\mathbf{L}_G$ and the vector $\mathbf{y}_G$ are defined by expanding Eq. (6-18), using (6-10), (6-11) and (6-16):

$$(\mathbf{L}_G)_{ij} = \int_a^b h_i(s)\,h_j(s)\,ds - \int_a^b \int_a^b h_j(s)\,K(s,t)\,h_i(s)\,ds dt, \qquad i, j = 1, \ldots, n. \tag{6-20}$$

$$\mathbf{y}_G = \int_a^b h_i(s)\, y(s)\, ds, \quad i = 1, \ldots, n. \tag{6-21}$$

6.2.3 Iterative Solution Methods

System (6-19) can be solved using an iterative algorithm such as a relaxation method. The presentation given here follows that of Gortler, Cohen and Slussalek [28], where relaxation methods are discussed in the context of global illumination. Let the vector of unknowns be **x**, and the matrix equation being solved be

$$\mathbf{Mx} = \mathbf{b}. \tag{6-22}$$

In a relaxation method, the approximation to the solution at step k is written $\mathbf{x}^{(k)}$. The residual at step k is given by the expression

$$\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{Mx}^{(k)}. \tag{6-23}$$

For a specific variable x_i, the residual is

$$r_i^{(k)} = b_i - \sum_j M_{ij} x_j^{(k)}. \tag{6-24}$$

A relaxation step sets the value of $x_i^{(k+1)}$ to be such that $r_i^{(k+1)}$ is zero. The Southwell relaxation algorithm selects the x_i to be the variable with the largest residual, $r^{(k)}$. Thus it is necessary to update all residuals at every iteration, but only one variable. The selected variable x_i is updated:

$$x_i^{(k+1)} = x_i^{(k)} + \frac{r_i^{(k)}}{M_{ii}}, \qquad r_i^{(k+1)} = 0. \tag{6-25}$$

All the residuals of the remaining variables are updated as follows:

$$r_j^{(k+1)} = r_j^{(k)} - \frac{M_{ji}}{M_{ii}} r_i^{(k)}, \quad j = 1, \ldots, n, \ \ j \neq i \tag{6-26}$$

It can be shown that if the system is row diagonally dominant or column diagonally dominant, the iterative solution method will converge (Gortler, Cohen and Slussalek [28]).

6.2.4 Previous Higher Order and Discontinuity Methods for Global Illumination

In previous discontinuity meshing algorithms (Lischinski, Tampieri and Greenberg [38] and [39], also presented in Tampieri [60]) the higher order interpolants created for display were not used during the solution process. Heckbert and Winget [35] proposed the framework for the combination of discontinuity meshing and higher-order methods, but results were presented only for two-dimensional environments (also presented by Heckbert in [34]). An extension to three dimensions was given by Heckbert in [33] but a complete Galerkin formulation was not presented.

On the other hand, higher order methods to date have not used discontinuity meshing (Gortler et al. [27], Zatz [72]). Troutman and Max [64] did not compute a discontinuity mesh, but did present a simpler framework similar to the formulation of the system presented below, using an user defined approximation to the discontinuity mesh. Zatz [72] presented a direct application of the Galerkin method, using orthonormal polynomials, on regular grids matched directly to the patches. He also presented a straightforward progressive refinement approach.

Our structured sampling algorithm draws on all the previous approaches in terms of the notation used. However, it is the first to fully incorporate discontinuity meshing into the solution process for a higher order solution method. In addition, the structured sampling algorithms developed in the previous chapters provide significant advantages over all the previous approaches in terms of expected accuracy and efficiency.

6.3 Formulating the Structured Global Illumination Method

In the structured method presented in the previous chapters, a mesh was built over which an interpolant approximation of radiance due to a single source was constructed. To introduce the formulation of the linear system, and the iterative solution method used, it will be assumed for the moment that the same fixed mesh is being used for all the iterations, and thus the same set of coefficients is being modified. This restriction will be lifted in Section 6.4, where the full benefits of the structured global solution will be realised.

6.3.1 Formulating the Linear System for Global Illumination

Let $\hat{B}_i(s, t)$ be the interpolant defined over the union of all domains Ω_i. If there are n basis functions, then

$$\hat{B}(s, t) = \sum_{i=1}^{n} b_i h_i(s, t) . \qquad (6\text{-}27)$$

Note that in reality, only the functions that are non-zero over each domain contribute to this summation, in that domain. These functions are defined in (s, t) space, which is a two-dimensional parameterisation of all the planar surfaces of each polygon in three dimensions. To overcome this notational difficulty, all of these domains are embedded in a hypothetical two-dimensional plane (this is equivalent to the construction of Heckbert and Winget [35] for two dimensions).

Let $\hat{E}_i(s, t)$ be the (known) function of emission over the surfaces in the environment, with $e_k, k = 1, \ldots, n$ being the corresponding coefficients of the basis functions. We are now in a position to write the system of equations for the Galerkin formulation of the global illumination problem. This construction is similar to that of Troutman and Max [64].

The equation for the computation of an approximation to radiance is given as follows:

$$\hat{B}(s, t) = \hat{E}(s, t) + \iint \hat{B}(u, v) K(s, t, u, v)\, du dv \tag{6-28}$$

Recall from Eq. (2-15), given the diffuse reflectance assumptions, that the *kernel* function $K(s, t, u, v)$ of the integral equation is defined as follows:

$$\begin{aligned} K(s, t, u, v) &= f_r(s, t, \theta_1, \theta_2)\, vis(u, v, s, t) \frac{\cos\theta_1 \cos\theta_2}{R^2} \\ &= \rho_i vis(u, v, s, t) \frac{\cos\theta_1 \cos\theta_2}{R^2}\,. \end{aligned} \tag{6-29}$$

Let Ω_i be the domain of h_i. Expanding equation (6-28) using the basis functions from (6-27), we have:

$$\sum_{k=1}^{n} b_k h_k(s, t) = \sum_{k=1}^{n} e_k h_k(s, t) + \iint_{\Omega_k} \left(\sum_{k=1}^{n} b_k h_k(u, v) \right) K(s, t, u, v)\, du\ dv. \tag{6-30}$$

By setting the residual to be orthogonal to each of the basis functions, we perform the step equivalent to that of Eq. (6-17), forming n equations:

$$\begin{aligned} \textstyle\sum_{k=1}^{n} b_k \iint_{\Omega_p} h_k(s, t)\, h_p(s, t)\, ds dt &= \textstyle\sum_{k=1}^{n} e_k \iint_{\Omega_p} h_k(s, t)\, h_p(s, t)\, ds\ dt \\ &+ \textstyle\sum_{k=1}^{n} b_k \iint_{\Omega_p} \iint_{\Omega_k} h_k(u, v)\, K(s, t, u, v)\, h_p(s, t)\, du\ dv\ ds\ dt, p = 1, \ldots, n \end{aligned} \tag{6-31}$$

Interestingly enough Eq. (6-31) does not directly correspond to the traditional equations used in the constant radiosity approximations such as that of Eq. (2-5) or (2-8). The index i varies across basis function space, rather than over surfaces or even elements. By setting:

$$K_{ij} = \iint_{\Omega_i} h_i(s,t) \left(\iint_{\Omega_j} h_j(u,v) K(s,t,u,v) \ du \ dv \right) ds \ dt, \tag{6-32}$$

and also

$$H_{ij} = \iint_{\Omega_i} h_i(s,t) h_j(s,t) \ ds \ dt, \tag{6-33}$$

$$E_i = e_i \iint_{\Omega_i} h_i(s,t) \ ds \ dt, \tag{6-34}$$

we get n equations

$$b_i - e_i = \sum_{j=1}^{n} b_j K_{ij}, \ \ i = 1, \ldots, n. \tag{6-35}$$

This system can be written in matrix form:

$$((\mathbf{H})_{ij} - (\mathbf{K})_{ij})\mathbf{B} = \mathbf{E} \Leftrightarrow \mathbf{MB} = \mathbf{E} \quad . \tag{6-36}$$

As explained above, the indexing in Eq. (6-36) is performed over the global numbering of basis functions and their coefficients. In is instructive to see how the elements of each matrix will be computed in practice.

For $\mathbf{H}_{ij}$, the only non-zero elements are those for which $\Omega_i \cap \Omega_j \neq \emptyset$. The domain Ω_i of each basis function h_i includes a set of neighbouring elements. These are all connected by the winged edge data structure. The neighbouring elements are found, and a list is formed containing the indices and the corresponding integrals associated with the basis function in question. The elements for vector $\mathbf{E}$ are similarly computed from Eq. (6-34). The matrix $\mathbf{K}_{ij}$ requires more computation, since it involves the geometric interactions between the different elements. In the iterative solution approach, only one specific column of this matrix will be computed at each iteration.

6.3.2 An Iterative Solution Method for Higher Order Global Illumination

The iterative method we will use is a modification of progressive refinement approach (Cohen et al. [16]), presented in terms of the Southwell relaxation method following Gortler, Cohen and

Slusallek in [28]. In that paper the progressive refinement approach for the constant radiosity linear system was cast as a Southwell relaxation method. We will be extending this approach for the solution of the matrix equation (6-36).

Progressive Refinement and Southwell Relaxation for Constant Radiosity-Based Systems

For the transformation of the constant radiosity system, Gortler, Cohen and Slusallek in [28] transform the system of equations slightly, so that the system is being solved for the "shot" radiosity B_i^s (i.e. radiosity that has been gathered at an element, and subsequently shot). The residual becomes the unshot radiosity B_i^u.

Following the presentation in Section 6.2.3, substitute the vector $\mathbf{x}$ in (6-22) by the vector $\mathbf{B}^s$, the forcing vector $\mathbf{b}$ by the emissivities $\mathbf{E}$ and also the vector $\mathbf{r}^{(k)}$ in (6-23) by the vector $\mathbf{B}^{u\,(k)}$. Also recall from Eq. (2-9) in Section 2.2.1, that the matrix $\mathbf{M}$ of equation (6-23) is given as follows for the constant radiosity-based method:

$$\mathbf{M} = \mathbf{M}^c = \begin{bmatrix} 1-\rho_1 F_{1-1} & \cdots & -\rho_1 F_{1-n} \\ & & \\ -\rho_n F_{n-1} & \cdots & 1-\rho_n F_{n-n} \end{bmatrix}. \tag{6-37}$$

The matrix equation thus becomes

$$\mathbf{M}^c\mathbf{B} = \mathbf{E}. \tag{6-38}$$

The iterative step to update the residuals is

$$B_j^{u\,(k+1)} = B_j^{u\,(k)} - \frac{M_{ji}^c}{M_{ii}^c} B_i^{u\,(k)} = B_j^{u\,(k)} + \rho_j F_{ji} B_i^{u\,(k)}, \tag{6-39}$$

which is exactly the progressive refinement update step Eq. (2-11).

Using Southwell Relaxation to Solve the Higher Order System

The above analysis gives a solid foundation for the progressive refinement method, and we extend it for the case in which the matrix $\mathbf{M}$ is given by the expression (6-36). To perform this, along with each coefficient b_i^s of the system (6-35), the residual b_i^u is stored, representing the contribution of this basis function to the radiance that has been collected but not shot from the elements overlapping the domain Ω_i.

At each iteration the variable i to be updated is chosen. For the constant radiosity system, the patch with the largest power is chosen. Since each iteration in the higher order system corresponds to the relaxation of a basis function coefficient, which represents only (a possibly partially negative) part of the radiance, such a choice can not be made based solely on the coefficient value. Instead, the element with the largest power is found.

If $l_N^e(s, t)$ is the interpolant approximating the unshot radiance function at this point in the solution process, the unshot power P_e^u of each element is simply

$$P_e^u = \iint_{\Psi_e} l_N^e(s, t)\, ds dt, \tag{6-40}$$

where Ψ_e is the geometric region over which the element is defined. The basis function with the largest coefficient, for which $\Omega_e \cap \Omega_i \neq \emptyset$ (i.e., domains overlapping the element), is chosen as the variable b_i^s to be relaxed. An alternative is to choose the basis function with the largest area of overlap. We will refer to the corresponding basis function h_i as the "shooting" basis function, and also to the corresponding domain Ω_i as the emitter domain, with the understanding that these terms do not correspond to the physical light transport.

Two operations are performed at each iteration. First, the "unshot" coefficient (i.e., the residual) is added to the "shot radiance" coefficient of i, and the unshot coefficient for i is set to 0, following equations (6-25) and (6-26):

$$b_i^{s\,(k+1)} = b_i^{s\,(k)} + b_i^{u\,(k)} / H_{ii}\ , \qquad b_i^{u\,(k+1)} = 0. \tag{6-41}$$

Note that $K_{ii} = 0$, because $K(s, t, u, v)$ in the inner integral of Eq. (6-32) is zero (since the cosines are zero) and thus $M_{ii} = H_{ii}$. Second, the residual update step is performed for all other basis functions:

$$b_j^{u\,(k+1)} = b_j^{u\,(k)} - \frac{M_{ji}}{M_{ii}} b_i^{u\,(k)} = b_j^{u\,(k)} + \frac{(H_{ji} - K_{ji})}{H_{ii}} b_i^{u\,(k)}\ , \quad j = 1, \ldots, n, \quad j \neq i. \tag{6-42}$$

Also, consider the set $S_1 = \{j_1, j_2, \ldots, j_m\}$ for which H_{ji} is non-zero (i.e., domains neighbouring i). For each $j \in S_1$, K_{ji} is zero, since these basis functions are also on the same surface. The set $S_2 = \{j_1', \ldots, j_w'\}$, are all the domains on surfaces other than that over which h_i is defined. For these $H_{ji} = 0$, since there is no overlap of basis function domains, and $K_{ji} \neq 0$, if the domain Ω_i is at least partially visible from Ω_j.

Discussion of Convergence

Gortler, Cohen and Slussalek [28], show how Southwell relaxation will converge if the matrix of the system is row or column diagonally dominant. Neither of these conditions is generally true for the matrix **M** of Eq. (6-36). As an alternative, we require the residual to decrease at each step, and will undo an iteration if this condition (which can be easily checked) does not hold.

For the system to converge, it is necessary that the total residual $\sum_j \left|b_j^u\right|$ decreases at every iteration. Since we use the Lagrange basis, which interpolates the radiance function, b_i^u must always be positive. To achieve reduction of the residual, we require:

$$\begin{aligned} \sum_j b_j^{u\,(k+1)} < \sum_j b_j^{u\,(k)} &\Leftrightarrow \sum_j b_j^{u\,(k)} - b_i^{u\,(k)} - \sum_{j,\, j \neq i} \frac{(H_{ji} - K_{ji})}{H_{ii}} b_i^{u\,(k)} < \sum_j b_j^{u\,(k)} \\ &\Leftrightarrow -b_i^{u\,(k)} \left(1 + \sum_{j,\, j \neq i} \frac{(H_{ji} - K_{ji})}{H_{ii}}\right) < 0 \\ &\Leftrightarrow -\left(\sum_{j,\, j \neq i} \frac{(H_{ji} - K_{ji})}{H_{ii}}\right) < 1 \quad , \end{aligned} \tag{6-43}$$

since we set the residual of variable i to zero (thus the subtraction of $b_i^{u\,(k)}$) and for every other residual $j, j \neq i$ we subtract $\left((H_{ji} - K_{ji}) / H_{ii} \right) b_i^{u\,(k)}$. Using the definitions of sets S_1 and S_2 given above, and since H_{ii} is always positive, we require the following to ensure residual reduction:

$$-\sum_{j,\, j \neq i} (H_{ji} - K_{ji}) < H_{ii} \Leftrightarrow \sum_{j \in S_2} K_{ji} - \sum_{j \in S_1} H_{ji} < H_{ii} \tag{6-44}$$

Condition (6-44) may or may not be satisfied at each solution step. The existence of higher order basis functions which can be negative, and the fact that each basis function is defined over a different geometric domain, makes it difficult to predict if (6-44) will hold at any given solution step. In general, it is expected to be true. The integrals K_{ji} are typically very small numbers, but may be either positive or negative. The integrals H_{ji} are also very small and often close to zero, and also positive or negative, while H_{ii} will tend to be larger since the integrand is always non-negative and interpolates unity at the node i. Thus it is extremely unlikely that the summation $\sum_{j \in S_2} K_{ji} - \sum_{j \in S_1} H_{ji}$ will be greater than H_{ii}. At each iteration, the condition (6-44) is checked.

This check can be performed very efficiently, since all the quantities have been computed. If it does not hold, the iteration is undone, and a different basis function selected. Note that convergence was not discussed by Troutman and Max in [64]. The Gauss-Seidel method they use also requires similar conditions to be met for the iterations to converge.

6.4 Solving Higher Order Global Illumination Using Structured Sampling

The formulation described in the previous section does not depend on the use of structured sampling, and thus could be used given any subdivision of an environment into a mesh over which interpolants are built. We show next how the structured sampling algorithms developed are particularly suited to the global solution. First, the complete structured sampling algorithm of Chapters 3-5 can be used to determine a good first estimate of the unshot radiance on a surface. Second, a modified version of the discontinuity mesh algorithm and the backprojection representation can be used to enhance the accuracy and the efficiency of the quadrature computations required for the iteration step of Eq. (6-42), and finally an adaptive mesh subdivision approach, based on the structured sampling algorithms, allows more accurate and more efficient approximations to radiance to be computed.

6.4.1 Constructing the Forcing Vector

The construction of the mesh of Lagrange polynomials in Chapters 3 through 5 was based on recognizing the structure of radiance functions, and generating interpolants for which tight error bounds exist. This approximation is accurate for two reasons. First, the actual geometry of the mesh elements is well suited to the behaviour of radiance over the surface in the environments, and second, the interpolants have been chosen based on the knowledge of the radiance function to achieve a good approximation. Thus the mesh represents the radiance from direct illumination well, and in general we expect little refinement to be required to represent the additional effects of secondary light transport in a diffuse environment. We show next how the structured sampling algorithm for direct illumination can also be seen as an approximation to the first iterations of the higher order method.

Assume that the structured sampling mesh has been constructed for a single light source, and that all coefficients of unshot radiance b_j^u are zero except on the light source. On the light source a constant basis function is assumed ($h_e = 1$), and $b_i^u = e$. If a Southwell iteration is to performed on the system (6-36) in this state, the update to the unshot radiance (residuals) on all domains other than that of the source, would be:

$$\begin{aligned} b_j^{u\,(1)} &= -\frac{(H_{ji} - K_{ji})}{H_{ii}} b_i^{u\,(0)} = -e_i \left(-\iint_{\Omega_j} h_j(u, v) \left(\iint_{\Omega_i} K(s, t, u, v)\, ds dt \right) du dv \right) \\ &= \left(\iint_{\Omega_j} L_S(u, v)\, h_j(u, v)\, du dv \right) \qquad j = 1, \ldots, n, \;\; \Omega_j \text{ not on light,} \end{aligned} \tag{6-45}$$

where H_{ji} is always zero since the light source contains only one constant element and no neighbours, and the inner integral is equal to the source to point form-factor of Eq. (2-26).

When we use structured sampling to construct the interpolants, we set

$$b_j^{u\,(0)} = L_S(u_i, v_j)\,, \tag{6-46}$$

where (u_j, v_j) are the coordinates of the node on element j. This is a reasonable approximation, since the basis function h_j has a value of 1 at (u_j, v_j) and goes to zero at the boundary of the domain (possibly becoming negative in between). In terms of convergence, the total residual has been significantly reduced since the total power levels have been dissipated into the environment and multiplied everywhere with the reflectance factors ρ_j that are all less than unity.

The constant light source shots are performed before starting the overall Galerkin process. In this way the set of n basis functions is defined by the mesh generated on each surface by the structured sampling algorithm and the "forcing" vector **E** in (6-36) is set up to have the values of the coefficients due to the constant light sources. The residual values b_i^u for each coefficient are then set equal to e_i^u, for $i = 1, \ldots, n$, on each domain.

6.4.2 Accurate and Efficient Quadrature Computation Using Discontinuity Meshing

For the computation of secondary illumination, a modified discontinuity computation is performed before the iterative step. Specifically, the domain Ω_i of the basis function h_i whose coefficient is being relaxed is used as the emitter polygon. A modified version of the discontinuity meshing algorithm of Chapter 4 is applied, using this polygon as the source.

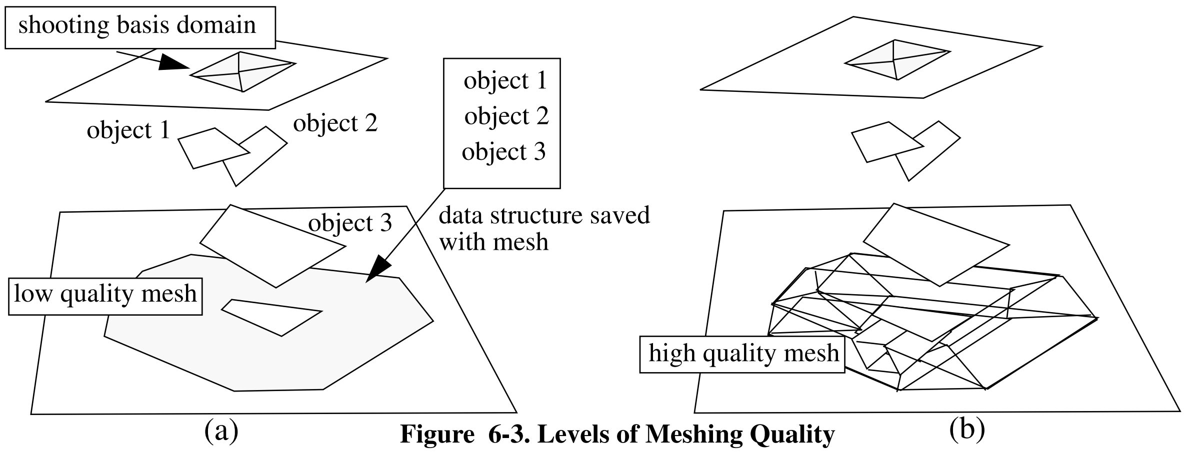


Figure 6-3. Levels of Meshing Quality

The discontinuity meshing algorithm can compute two levels of mesh quality for secondary illumination calculations. The lower quality mesh includes only the penumbral boundaries (formed only by maximal emitter EV events), and an approximation to the umbra/penumbra boundary (Figure 6-3(a)). This approximation calculates all the minimal emitter-EV events, and a convex polygonal approximation to the full boundary that would include EEE events. The high quality version is the complete discontinuity mesh (Figure 6-3(b)). Some heuristics for the choice of the quality of mesh construction are outlined in Section 6.5. Extensive testing needs to be performed to determine the effectiveness of such choices.

The algorithm starts by computing the low-quality mesh, and stores all intermediate objects encountered when generating the umbral and penumbral boundaries. Thus the mesh can be refined efficiently to become the complete mesh, when such refinement is required (see Figure 6-3). This is similar in spirit to the blocker lists maintained by Teller and Hanrahan [61].

For each basis function h_j the required integration is:

$$K_{ij} = \iint_{\Omega_j} h_j(u, v) \left(\iint_{\Omega_i} h_i(s, t)\, K(s, t, u, v)\, ds dt \right) du dv \tag{6-47}$$

This integral is computed using a product quadrature rule, typically Gauss or Gauss-Legendre (Zatz [72], Schroeder [52]). This is written, following the notation in Zatz [72], where $(u^{\lambda}, v^{\mu}) \in \Omega_j$ $(s^{\iota}, t^{\kappa}) \in \Omega_i$, are Gauss points and w_i^{ι} are the weights:

$$K_{ij} \approx \sum_{\iota, \kappa, \lambda, \mu} K(s^{\iota}, t^{\kappa}, u^{\lambda}, v^{\mu})\, h_i(s^{\iota}, t^{\kappa})\, h_j(u^{\lambda}, v^{\mu})\, w_i^{\iota} w_i^{\kappa} w_j^{\lambda} w_j^{\mu}. \tag{6-48}$$

For both high and low quality, the element domains Ω_j that are completely visible from the "shooting" basis function Ω_i are identified. For these, the visibility term $vis(s, t, u, v)$ normally needed for each evaluation of $K(s, t, u, v)$ (see Eq. (6-29)) is not required (see Figure 6-4(a), left). This allows a large number of the numerical integrations of each iteration to be performed very efficiently and accurately, since the functions $h_i(s, t)$ are simple polynomials.

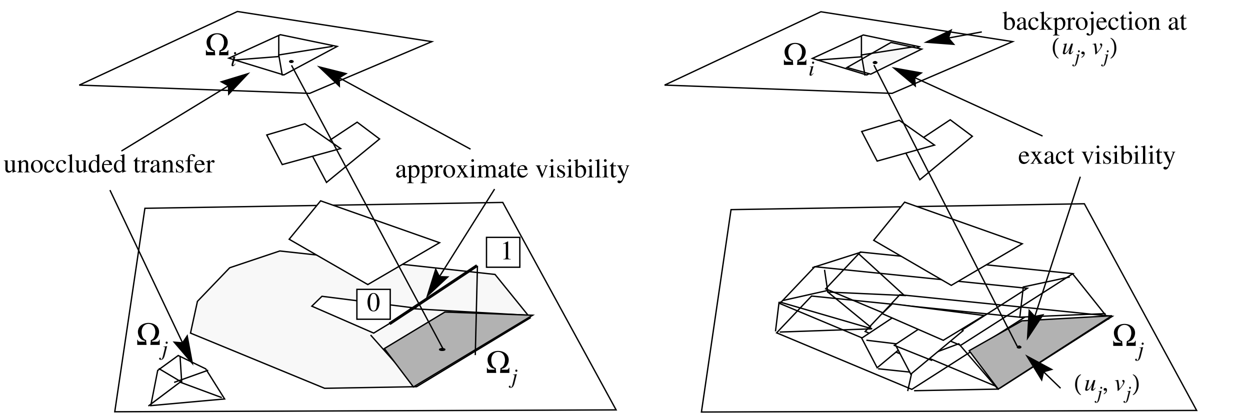


Figure 6-4. Accurate and Efficient Quadrature

For the case in which the full discontinuity mesh is computed, assume that the element Ω_j is partially occluded with respect to Ω_i, and thus the backprojection at any point (u_j, v_j) can be efficiently computed. For a given evaluation $K(s^{\kappa}, t^{\lambda}, u^{\mu}, v^{\nu})$, the backprojection can be determined at (u^{λ}, v^{μ}), and the corresponding point $(s^{\kappa}, t^{\lambda})$ on the "shooting" domain, can be immediately determined to be visible or not (see Figure 6-4(b)). Thus, the backprojections allow very efficient evaluation of the visibility function inside the kernel.

When the low-quality meshing is used, an approximation is made by assuming a visibility function $VisApprox(s, t, u, v) \approx vis(s, t, u, v)$, that is 1 when (u, v) is on the penumbra/light border, and zero when it is on the umbra/penumbra border (see Figure 6-4(a), right). This approximation is similar in concept to that of shadow masks used by Zatz [72]. His approach however did not include a discontinuity mesh, and was a selective heuristic approximation, applied only to limited cases.

Compared to previous approaches the structured method will be significantly more accurate and efficient. In all previous cases visibility had to be tested for each evaluation of the kernel function in Eq. (6-47). This testing, either with ray-casting (as in Gortler et al. [27], or z-buffer operations in Troutman and Max [64]), is inaccurate, subject to the inevitable problems of point-sampling, and expensive since the entire environment has to be examined for function evaluation at each point. In contrast, even in the case of the low-quality mesh, the expected error in visibility calculations is minimal, and of course for the entire iteration only one set of visibility operations is performed.

6.4.3 Adaptive Mesh Modification

The previous discussion implicitly assumes that the mesh defining the system of Eq. (6-35) is invariant at each iteration. Using the structured sampling algorithms we can adaptively modify the mesh at each step, thus achieving a better representation of radiance as light is distributed over the environment.

A discontinuity mesh of high or low quality is calculated at each iteration, once the “shooting” basis function has been chosen. As described in the previous section, this mesh is used to perform efficient quadrature evaluations. In many cases, it will not be further required, and it can be discarded. In others, it will be used to adaptively subdivide the previously existing mesh. In addition, for the regions that are unoccluded with respect to the shooting element, the structured approach of Chapter 3 can be used as an approximation to determine whether subdivision is required. The cases for which further subdivision is performed are outlined in Section 6.5.

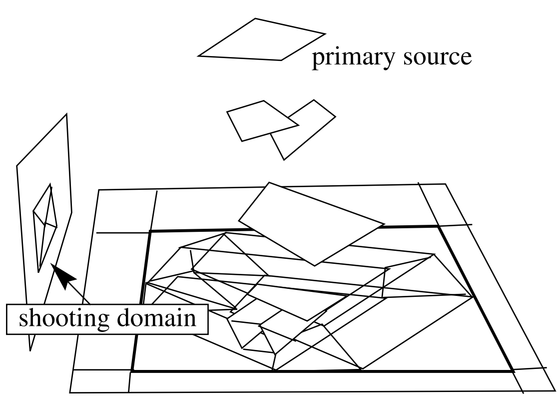
Consider an iteration step k, at which h_i has been chosen as the emitter. Let S_j be a surface for which we have decided that adaptive subdivision is required. Assume that the surface S_j is completely unoccluded with respect to the emitting domain, and that a mesh M_j is associated with the surface, containing n_j nodes with unshot radiance coefficients b^u_m, $m \in \{i_1, \ldots, i_{n_j}\}$. The polynomial interpolant defined by the basis function and the coefficients $b_m^{u(k)} - b_m^{u(k-1)}$ (i.e., the effect of iteration k on the interpolants) is related to the radiance that would be emitted from the domain Ω_i if it was constant diffuse emitter, with luminosity equal to the value $b_i^{u(k-1)}$. This is true since the Galerkin method in essence simply weights the radiance by the basis functions. Thus, a first fast approximation to the effect of a shot on the basis function can be performed by calculating the estimated values $b^e_m, m \in \{i_1, \ldots, i_{n_j}\}$ of radiance at the desired nodes. These b^e_m are computed using Eq. (2-26), where the luminosity M is set equal to b^u_i and the domain Ω_i is considered a constant diffuse light source.

Using this approximation, the unoccluded regions can be segmented by applying the structured sampling algorithm of Chapter 3. This approximation gives a rough idea of how much the iterative step will affect the interpolant function built on the receiving surface and allows the mesh to be adaptively modified to better approximate this change.

6.4.4 Hierarchical Transfer

From the form of equations (6-32) to (6-35) and (6-42), it can be seen that the integrations perform multiple levels of averaging of the radiance function as it varies over the emitter domain and the receiver domain.

It is thus possible that a transfer from the shooting basis function to a penumbral domain constructed from previous iterations (usually due to direct illumination), containing many elements, may be adequately approximated by performing the transfer to a coarser mesh. This coarse mesh is defined over the bounding box of the penumbral domain.



Once a domain has been identified as suitable for the coarse-level transfer, the underlying penumbral elements are sampled at the nodes of the coarser mesh. The iteration is thus performed on a simpler system, in which coarse level basis functions and coefficients are used instead of the more numerous penumbral domains. Thus the number of residual updates and quadrature evaluations will be significantly reduced. If the resulting coefficients show large differences in value, indicating that the coarse level transfer was insufficient, the iteration is refined onto all the remaining underlying domains, and the coarse level shot is discarded. Otherwise, each node of the penumbral domain evaluates the resulting interpolant defined on the coarse mesh and adds the difference into the coefficients.

6.5 Additional Observations and Heuristics

In the general framework laid out in the previous section, several heuristic algorithms are required to determine when a lower quality solution can be applied. The proposed methods are a starting point which require experimental verification. Some details of these heuristics are presented next. Additional observations are also made concerning the structured global illumination methodology.

6.5.1 Choosing Discontinuity Meshing Quality and Adaptive Meshing

For most of the secondary illumination light transfer, it is expected that low-quality meshing will be sufficient. The choice will be based on whether the detailed backprojection information and discontinuity meshing will make a significant difference in the resulting image.

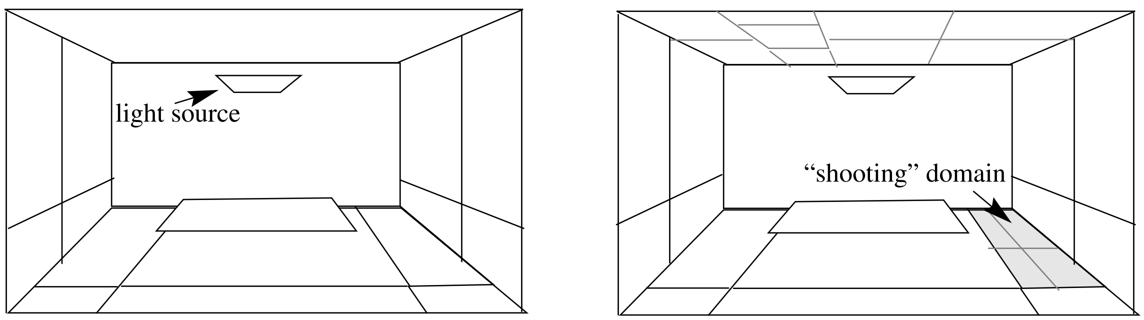
Once the coarse discontinuity mesh has been computed for a surface S_j, the estimated values of the basis function coefficients b_m^e are examined at the n_j nodes on the coarse mesh boundary of penumbra and light, using the approximation discussed in Section 6.4.3. In addition the value of the existing interpolant $b_m^p = b_m^u + b_m^s$ is queried at these points. Two conditions are examined.

1. We first determine if $b_m^p > b_m^e$ and the values of $\left|b_m^p - b_m^e\right|$ are less than a certain tolerance, or $b_m^p < b_m^e$, for $m \in \{i_1, \ldots, i_{n_j}\}$.
2. We determine if b_m^e, converted to pixel RGB values, is greater than a certain tolerance.

If both conditions 1 and 2 are met, the complete discontinuity mesh is calculated.

To determine when adaptive meshing is required for an unoccluded region, the position of the maximum is found, if it exists, using the structured sampling technique for the approximation of radiance described in Section 6.4.3. In addition, the nodes on the boundary of the unoccluded regions are examined. If the conditions 1 and 2 presented above are true for all the nodes in question, then adaptive subdivision of this unoccluded region is performed.

The discontinuity mesh is used to segment the receiving surface if conditions 1 and 2 are satisfied, but for a smaller tolerance. A coarse discontinuity mesh is created when the "shot" is the first light transfer to occur onto the surface, even if the power levels are very low. For example, the ceiling is assigned a mesh in Figure 6-5, even though the effect of the shooting is minimal. For subsequent shots, it is possible that this mesh will be sufficient.



Coarse view of mesh from direct illumination; notice lack of mesh on ceiling

A mesh is assigned to the ceiling even though the power levels are low

Figure 6-5. Adaptive Mesh Subdivision

6.5.2 Singular Transfers and Domain Consistency

As observed by Zatz [72], the kernel function will be singular when a receiver and an emitter share a common edge. There have been several solutions to this problem presented by Zatz [72] and also by Schroeder [52]. The methodology of Schroeder can be adapted to our solution, by designing specialised quadrature rules that deal with the singularity.

To maintain consistency, all T-vertices remaining in the mesh constructed by the structured method must be removed, to allow the construction of domains suitable for the Galerkin formulation. To do this without unnecessarily splitting the tensor products, a set of triangular elements is placed around the penumbral group where required.

6.6 A Structure-Driven Global Illumination Algorithm

All the steps described above constitute the elements of a structured global illumination algorithm. The convergence criterion comes naturally from our definition of residuals. The system has converged when either the total unshot power in the system has dropped below a certain level or the maximum unshot power of an individual element has fallen below a certain level. Let $E = \{e_1, \ldots, e_m\}$ be the set of all elements in the scene. Then convergence is reached when:

$$\sum_{e=1}^{m} P_e^u < \text{TotalTolerance}, \quad \text{or} \qquad \max_{e \in E} (P_e^u) < \text{SingleTolerance} \tag{6-49}$$

The structure-driven global illumination algorithm starts with the computation of the mesh due to the light sources, using the structured algorithm of Section 5.6. Once this step is complete, and the meshes have been merged, the iterative solution process can begin. The "shooting" basis function is chosen, and the corresponding discontinuity mesh of appropriate quality is computed on each surface visible from the emitting domain. For each such surface, the existing mesh is adapted if required, and it is decided whether a coarse or fine level transfer will be performed between the "shooting" domain and each region of the receiving surface. At this point the system of Eq. (6-36) can be formed. An iteration is performed, the convergence criterion is checked, and the solution continues until convergence. After convergence, shot and unshot coefficients are added to form the final coefficient values. This procedure is shown in the pseudocode of Figure 6-6.

```
global_structured_sampling( Scene s )
{
    Mesh mp = ∅;
    foreach source i in light sources do
          Mesh mi = complete_structured_sampling(s, i);
          mp = merge_meshes( mi, mp );
    done
    while( !converged )do
          BasisFunction bi = choose_basis_function();
          Mesh gm = compute_mesh_for_secondary_shot( bi );
          foreach visible surface j do
                 adapt_mesh( gm , bi, j );
                 choose_transfer_quality ( bi, j );
          done
          BasisFunction bj;
          retrieve Hii from bi
          foreach bj on visible surfaces do
                 compute Kji and get Hji where appropriate
                 bj_u = bj_u + bi_u*(Hji + Kji) / Hii
          done
          if( !( Σ_{j∈S2} Kji − Σ_{j∈S1} Hji < Hii ) ) undo iteration
          else
                 bi_s = bi_s + bi_u/Hii; bi_u = 0;
    done
    foreach basis function i do
          bi_s = bi_s + bi_u;
}
```

$$\sum_{j \in S_2} K_{ji} - \sum_{j \in S_1} H_{ji} < H_{ii}$$

Figure 6-6. A Structure-Driven Global Illumination Algorithm

6.7 Discussion

In this chapter a general framework was presented for an iterative solution to a Galerkin formulation of the global illumination problem.

The mesh of triangular and tensor product Lagrange basis functions constructed by the structured sampling algorithm was used to initiate the global solution. An algorithm for merging meshes from multiple sources was presented.

The formulation of the iterative solution method for global illumination was detailed, and conditions for convergence of the system were derived. It was shown that the structured algorithms generalise to the global illumination problem and provide a more efficient and accurate solution. These advantages are summarised below.

A good initial approximation is achieved using the mesh generated by the structured sampling algorithm, applied for multiple primary constant diffuse sources. This approximation is used to initialise the set of domains for the basis functions as well as the unshot radiance coefficients for all the domains in the scene.

At each iteration, discontinuity meshing is performed at a high or a low level of quality, based on the requirements of each interaction between emitter and receiver. For both levels, the expensive visibility computation within the kernel of the integral equation is eliminated for mutually visible domains. For domains that are partially visible, the backprojections computed by the meshing algorithm allow exact samples of the kernel function to be computed, enhancing accuracy and efficiency of the quadrature rule. Low quality meshing also provides an approximation to visibility that avoids point sampling problems and is significantly more efficient, compared to previous methods.

The structured algorithm facilitates adaptive mesh modification before each iteration, again enhancing the quality of the solution. Finally, the light transfer can be efficiently computed at a coarse level and if necessary subsequently computed at a finer level, without additional expense.

Through this formulation, we have shown that the structured sampling ideas generalise easily to the global illumination problem, resulting in an algorithm that shows significant advantages over previous approaches. It is expected that an implementation of the general global solution will show significantly improved convergence rates, and will also allow the evaluation of the proposed heuristics for quality selection.

Chapter 7
Conclusions

In this thesis a new approach has been presented for the sampling and reconstruction of radiance in diffuse polyhedral environments. This approach is based on the study and identification of important properties of radiance structure. This study, performed for unoccluded and occluded environments, in conjunction with the fast discontinuity meshing algorithm developed in this thesis, has led to the introduction of accurate and efficient sampling and reconstruction algorithms for radiance.

The result of these algorithms is the construction of a sparse and accurate interpolating approximation to radiance in diffuse scenes with occlusion. These interpolants are used to produce high-quality renderings of such scenes, as well as interactive walkthroughs using suitable polygonalisation. Finally the algorithms developed allow the formulation of a Galerkin-type finite element method to solve the global illumination problem. In this concluding chapter, the results of the thesis are briefly summarised, and future directions of research are discussed. We conclude the thesis with some final thoughts.

7.1 Summary of Results

The research in this thesis was structured as follows. In Chapter 3 unoccluded illumination was treated, while in Chapter 4 the computation of penumbral boundaries was detailed. Sampling and reconstruction in the penumbrae was presented in Chapter 5 and in Chapter 6 the structured approach was generalised to global illumination. The results are summarised for each chapter.

7.1.1 Chapter 3: Structured Sampling for Unoccluded Area Light Sources

Radiance on an arbitrarily oriented surface due to a diffuse disc light source was proven to be unimodal. Unimodality was then conjectured to hold for convex light sources. Based on these observations, a structured sampling algorithm was developed. This algorithm finds the maximum of radiance and subdivides based on curvature.

Adaptive subdivision was performed that includes built-in error bounds, due to the segmentation of the radiance function based on its maximum and its curvature. Interpolants representing radiance were built based on this subdivision. The results of the implementation show that the structured sampling algorithm performs significantly better, both numerically and visually, compared to uniform subdivision with the same number of samples, as well as displaying consistency in animation.

7.1.2 Chapter 4: Computing Penumbral Boundaries

A fast algorithm was developed that computes the complete discontinuity mesh. The implementation of this algorithm has resulted in the computation of the complete mesh for the first time for interesting computer graphics environments. The algorithm capitalises on observation of geometric structure in the scene and on the properties of the resulting mesh to perform the mesh calculation efficiently. Extensive use of voxel grid spatial subdivision further enhances the efficiency of the solution.

Within each face of the discontinuity mesh the σ-aspect is the same. The backprojection is an abstract representation of the visibility of the source, and it is used to efficiently compute the visible portions of the source at any point in a face. Thus exact radiance values at every pixel can be efficiently computed for images with occlusion. An incremental algorithm is presented for the calculation of the backprojections, that allows their computation to be performed without the need to query the environment. This drastically reduces the cost of computing exact illumination in scenes with occlusion, compared to previous approaches.

7.1.3 Chapter 5: Structured Sampling for Scenes with Occlusion

From empirical studies it is observed that radiance in the penumbrae is monotonic in the majority of the faces of the discontinuity mesh. The properties of radiance in the remaining faces are investigated and the major factors effecting this function behaviour are studied and listed. These properties are used to construct a triangulated domain over which low-order interpolants are constructed for radiance.

An algorithm is presented that removes a significant fraction of the discontinuity mesh edges without significant degradation in the quality of approximation. Similarly the degree of interpolation required on each edge of the triangulation is determined to be linear or quadratic, depending on the variation of radiance along these edges.

The structured sampling algorithms for the unoccluded and the occluded case are then combined to construct an interpolant for radiance for general environments with occlusion. The results of the implementation show that the approximation is of high quality and that both the mesh reduction and the degree selection can be performed without significant quality degradation.

7.1.4 Chapter 6: Extending Structured Sampling for Global Illumination Calculations

A mathematical framework is presented in which the structured approach is used for a Galerkin-type iterative solution to the global illumination problem. The algorithms developed for structured sampling of direct illumination facilitate more efficient and more accurate quadrature calculations, and iterative adaptive mesh construction. Heuristics are also presented for efficient adaptive quality visibility and light transport calculations. It is expected that this approach will display faster convergence and higher quality than previous methods.

7.2 The Future

In this thesis, the concept of structured sampling for illumination calculations has been introduced. It has led to the development of efficient and accurate sampling and reconstruction algorithms for illumination functions. This approach has yielded significant results, and there are numerous possibilities for extensions and refinement for future work. In this section, promising future directions are discussed.

Unoccluded Illumination

A formal proof of Conjecture 3-1 is desirable. Another interesting path of research is the determination of the largest class of sources which emit unimodal radiance functions, and also a set of criteria that will allow the determination of inclusion in this class.

The expense of finding the maximum and the inflection points for large environments is higher than desirable. Approximations to these calculations should be investigated. In particular, two disc sources can be used, one bounding and one bound by the polygonal source. The average of the maxima of the two discs may prove to be very close to the maximum of the polygonal source itself. Also, for objects lying in the same plane, the maximum computation should be performed only once. For parallel surfaces, or almost parallel surfaces, an algorithm can be found so that the maximum and inflection point computations need only be performed for one surface. This computation will then be used to compute an approximation for these calculations on all the related surfaces. For scenes including many objects such as chairs, desks, shelves etc., many of the surfaces are parallel or close to parallel (typically in vertical or horizontal orientations), rendering such an approach very beneficial.

The nature of interpolants required to reconstruct unoccluded radiance must also be investigated at length, especially with respect to visual artifacts that are produced by different degrees or basis function families. An interesting route is to perform a similar investigation of radiance behaviour for non-constant, non-diffuse environments, determining the different properties for light sources with more interesting emitter properties and for specular or even more complex (e.g., anisotropic) reflectors.

Computation of Penumbral Boundaries

Extensive statistics for geometric structure should be gathered on large varieties of scenes. This will allow the determination of scene properties for more comprehensive classes of scenes. Interesting properties that are important for the prediction of running time of the meshing algorithm include the average size of the discontinuity mesh with respect to scene size, emitter size etc., the ratio of the number of important EEE surfaces vs. the number of important EV surfaces, the average variation of radiance within penumbral groups, to mention a few.

These statistics will aid in our understanding of the geometric visibility properties for a much larger class of computer graphics environments. This will allow us to determine when our meshing algorithm will perform well, or when an algorithm which has better asymptotic complexity behaviour is necessary, such as that of Stewart and Ghali [56].

Reconstruction of Radiance in Penumbral Regions

As with the study of geometric structure, extensive studies of different classes of scenes need to be performed, to determine the general behaviour of radiance within the penumbra. More extensive statistics will result from such studies, giving a more concrete understanding of radiance behaviour in the penumbrae in more general configurations. This will hopefully lead to further improvements in the domain construction and interpolant selection algorithms for such scenes.

More comprehensive and general statistics may lead to the development of predictive algorithms that will determine which discontinuity surfaces need to be cast, thus significantly reducing the expense of computing the discontinuity mesh.

Incremental Algorithms for Animation

The nature of our discontinuity meshing algorithm allows local changes to be made that do not affect the rest of the mesh. Such local updates are useful in animations in which a small object moves, and thus potentially only a small part of the mesh needs to be modified. The voxel spatial subdivision structure can be used extensively to allow such updates to be performed efficiently.

The algorithm for mesh reduction suffers from lack of consistency across animated sequences. With the development of the incremental algorithm for mesh updates, certain criteria can be applied to the mesh reduction approach to ensure consistency over animation, with respect to the edges removed from the mesh.

Global Illumination

A full implementation of the global illumination solution is necessary to evaluate all the aspects of the new higher-order iterative method proposed in Chapter 6.

The mesh and interpolant merging heuristics proposed must be subjected to extensive numerical testing. The results of such numerical experimentation will inevitably lead to better criteria for mesh and interpolant merging. Similarly the criteria determining when to use the interpolant hierarchy will evolve based on the results of experimentation.

The criteria for varying the quality of the visibility calculations and the interpolants used to represent illumination over the surfaces in the environment must also be subjected to rigorous testing procedures. It is expected that in many cases the lower quality visibility and transport calculations will be sufficient for secondary illumination of diffuse scenes.

Finally, suites of test configurations must be run using the implementation of the new approach to determine the overall factors that affect convergence and error. It is expected that the use of the structured approach will lead to faster convergence, since major sources of error are greatly reduced, and the largest part of unshot power is processed early in the solution process.

Clustering

The ultimate solution to global illumination hinges on grouping and hierarchical techniques for very complex scenes, for which it is necessary to achieve sub-linear processing times. The ideas developed in this thesis could potentially be extended to accommodate hierarchies of objects, considering the properties of illumination from the group as a whole, similar in spirit to the FIAT algorithm proposed by Fournier et al. [22].

The iterative higher order method would be extended to represent illumination passing through "walls" of bounding volumes of complex objects. A projection mechanism from the higher geometric detail to the coarser bounding box representation of radiance must be determined, together with an inverse mapping.

Global Illumination for General Environments

The structured sampling methodology can be applied to the study of more general environments. For these, properties similar to unimodality and monotonicity determined in the diffuse case must be found. Algorithms based on the identification of such properties can then take advantage of these properties to efficiently and accurately simulate illumination and light transport in such environments.

7.3 Last Thoughts

The basic approach taken in this thesis was to tackle the simpler direct illumination problem for which careful and meaningful analysis can be performed, instead of the more complicated global illumination problem. This approach has proven to be successful. Much has been learned about the structure of light in non-trivial configurations, and the algorithms developed from these observations have given successful results, and are easily generalisable to the global illumination problem.

Looking back, it is evident that the immense complexities of solving the many problems for the supposedly "simple" case, and the size and difficulties of the resulting implementation were drastically underestimated. On the positive side, these solutions and implementations have advanced our understanding of many important factors concerning illumination calculations and provided new and promising algorithms.

Finally, from the experience attained performing this research, it is evident that the solution to the problem of global illumination calculations entails more than meets the eye, and that the goal of real-time, fully accurate global illumination solutions for general computer synthesized environments is still far in the future. Hopefully, we have made one small step forward with this thesis.

References

[1] Amanatides, John, "Ray Tracing with Cones," *ACM Computer Graphics (SIGGRAPH '84 Proceedings)*, vol. 18, no. 3, July 1984.

[2] Amanatides, John and Andrew Woo, "A Fast Voxel Traversal Algorithm for Ray Tracing," *Proceedings of Eurographics '87*, Elsevier Science Publishers B.V.(North-Holland) 1987.

[3] Appel A., "Some Techniques for Shading Machine Renderings of Solids," *Proceedings of AFIPS JSCC*, vol. 32, pp. 37-45, 1968.

[4] Atherton P., K. Weiler and Donald P. Greenberg, "Polygon Shadow Generation," *ACM Computer Graphics (SIGGRAPH '78 Proceedings)*, vol. 12, no. 3, July 1978.

[5] Aupperle Larry, and Pat Hanrahan, "Importance and Discrete Three Point Transport," *4th Eurographics Workshop on Rendering*, Paris, France June 1993.

[6] Baum, Daniel R., Holly E. Rushmeier and James M. Winget, "Improving Radiosity Solutions Through the Use of Analytically Determined Form-Factors," *ACM Computer Graphics (SIGGRAPH '89 Proceedings)*, vol. 23, no. 3, July 1989.

[7] W. J. Bouknight and K. Kelley, "An Algorithm for Producing Half-Tone Computer Graphics Presentations with Shadows and Movable Light Sources," *Proceedings of SJCC, AFIPS*, vol. 36, 1970.

[8] Campbell, A. T. III and Donald S. Fussel, "Adaptive Mesh Generation for Global Diffuse Illumination," *ACM Computer Graphics (SIGGRAPH '90 Proceedings)*, vol. 24, no. 4, August 1990.

[9] Campbell, A. T. III and Donald S. Fussell, "An Analytic Approach to Illumination with Area Light Sources," *Technical Report TR-91-25, Computer Science Dept, University of Texas at Austin*, August 1991.

[10] Chen, Shenchang Eric, "A Progressive Radiosity Method and its Implementation in a Distributed Processing Environment," *Masters of Science Thesis*, Cornell University, January 1989.

[11] Chen, Shenchang Eric, Holly E. Rushmeier, Gavin Miller and Douglass Turner, "A Progressive Multi-Pass Method for Global Illumination," *Computer Graphics (SIGGRAPH '91 Proceedings)*, vol. 25, no. 4, pp. 164-74, July 1991.

[12] Chin, Norman and Steven Feiner, "Near Real-Time Shadow Generation for Global Diffuse Illumination," *ACM Computer Graphics (SIGGRAPH '90 Proceedings)*, vol. 24, no. 4, pp. 99-106, August 1990.

[13] Chin, Norman and Steven Feiner, "Fact Object Precision Shadow Generation for Area Light Source using BSP Trees," *ACM Computer Graphics (SIGGRAPH Symp. on Interactive 3D Graphics)*, March 1992.

[14] Cohen, Michael F. and Donald P. Greenberg, "The Hemi-cube, A Radiosity Solution For Complex Environments," *ACM Computer Graphics (SIGGRAPH '85 Proceedings)*, Vol. 19, No. 3, July 1985.

[15] Cohen, Michael F., Donald P. Greenberg, David S. Immel, and P. J. Brock, "An Efficient Radiosity Approach for Realistic Image Synthesis," *IEEE Computer Graphics & Applications*, vol. 6, no. 3, March 1986.

[16] Cohen, Michael F., Eric S. Chen, John R. Wallace and Donald P. Greenberg, "A Progressive Refinement Approach to Fast Radiosity Image Generation," *ACM Computer Graphics (SIGGRAPH '88 Proceedings)*, vol. 22, no. 4, August 1988.

[17] Christensen Per H., David H. Salesin and Tony D. DeRose, "A Continuous Adjoint Formulation for Radiance Transport," *4th Eurographics Workshop on Rendering*, Paris, France, June 1993.

[18] Delves L.M. and J. L. Mohammed, "*Computational methods for Integral Equations*", Cambridge University Press, Cambridge U.K., 1985.

[19] Drettakis, George and Eugene L. Fiume, "Accurate and Consistent Reconstruction of Illumination Functions Using Structured Sampling," *Proceeding of the Eurographics Conference*, Barcelona, Spain, September 1993.

[20] Drettakis, George and Eugene L. Fiume, "Concrete Computation of Global Illumination Using Structured Sampling," *3rd Eurographics Workshop on Rendering*, Bristol, UK, May 1992.

[21] Franklin W. R., "A Linear Time Exact Hidden Surface Algorithm", *ACM Computer Graphics (SIGGRAPH '80 Proceedings)*, vol 14, no. 3, pp. 117-123, July 1980.

[22] Fournier, Alain, Eugene Fiume, Marc Ouellette, and Chuan K. Chee, "FIAT LUX," *DGP Technical Memo 90-1*,January 1990.

[23] Glassner, Andrew S., "Maintaining Winged-Edge Models," In "*Graphics Gems II*", edited by Jim Arvo, Academic Press Inc., 1991.

[24] Gouraud, Henri, "Continuous Shading of Curved Surfaces," *IEEE Transactions on Computers*, vol. c-20, no. 6, June 1971.

[25] Gigus, Ziv and Jitendra Malik, "Computing the Aspect Graph for the Line Drawings of Polyhedral Objects," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 12, no. 2, pp. 113-133, February 1990.

[26] Gigus, Ziv, John Canny, and Raimund Seidel, "Efficiently Computing and Representing Aspect Graphs of Polyhedral Objects," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 13, no. 6, pp. 542-551, June 1991.

[27] Gortler, Steven J., Peter Schroeder, Michael F. Cohen, and Pat Hanrahan, "Wavelet Radiosity," *ACM Computer Graphics (SIGGRAPH '93 Proceedings),* August 1993.

[28] Gortler Steven, Michael F. Cohen and Phillip Slusallek, "Radiosity and Relaxation Methods: Progressive Refinement is Southwell Relaxation", *TR-93-408* CS Dept. Princeton University 1993.

[29] Goral, Cindy M., Kenneth E. Torrance, Donald P. Greenberg, Bennet Battaile, "Modelling The Interaction Of Light Between Diffuse Surfaces", *ACM Computer Graphics (SIGGRAPH '84 Proceedings)*, vol 18, No 3, July 1984.

[30] Hanrahan, Pat, "How To Solve It? The Scientific Basis of Rendering," Invited Talk, *Eurographics '93 Workshop on Rendering*, Paris, France, June 1993.

[31] Hanrahan Pat, and David Salzman, "A Rapid Hierarchical Radiosity Algorithm for Unoccluded Environments," *Proceedings of the Eurographics Workshop on Photosimulation, Realism and Physics in Computer Graphics*, Rennes, France, July 1990.

[32] Hanrahan, Pat, David Salzman, and Larry Aupperle, "A Rapid Hierarchical Radiosity Algorithm," *ACM Computer Graphics (SIGGRAPH '91 Proceedings),* vol. 25, no. 4, pp. 197-206, July 1991.

[33] Heckbert, Paul, "Discontinuity Meshing for Radiosity," *3rd Eurographics Workshop on Rendering*, Bristol, UK May 1992.

[34] Heckbert, Paul, "Radiosity in Flatland," *Proceedings of Eurographics '92*, Cambridge (UK), Elsevier Science Publishers B.V. (North-Holland) 1992.

[35] Heckbert Paul, and James M. Winget, "Finite Element Methods for Global Illumination," EECS/University of California Berkeley, Report No. UCB/CSD 91/643, July 1991.

[36] Kahaner David, Cleve Moler and Stephen Nash, "*Numerical Methods and Software*", Prentice Hall Series in Computational Mathematics, New Jersey, 1989.

[37] Kajiya, James T., “The Rendering Equation”, *ACM Computer Graphics (SIGGRAPH ‘84 Proceedings)*, vol 20, no 4, August 1986.

[38] Lischinski, Dani, Fillipo Tampieri, and Donald P. Greenberg, “Combining Hierarchical Radiosity and Discontinuity Meshing,” *ACM Computer Graphics (SIGGRAPH ‘93 Proceedings)*, vol. 27, no. 4, August 1993.

[39] Lischinski, Dani, Fillipo Tampieri and Donald P. Greenberg, “Discontinuity Meshing for Accurate Radiosity,” *IEEE Computer Graphics & Applications*, vol. 12, no. 6, pp. 25-39, November 1992.

[40] Moon, Parry, The Scientific Basis of Illuminating Engineering, *McGraw-Hill*, 1936.

[41] Nishita, Tomoyuki and Eihachiro Nakamae, “Half-tone Representation of 3-D Objects Illuminated by Area Sources or Polyhedron Sources”, *Compsac, Proc. IEEE 7th Intl. Comp. Soft and Applications Conference*, pp. 237-242, November 1983.

[42] T. Nishita and E. Nakamae, “Continuous Tome Representations of Three-Dimensional Objects Taking Account of Shadows and Interreflections,” *ACM Computer Graphics (SIGGRAPH ’85 Proceedings)*, vol. 19, no. 3, July 1985.

[43] Pratt, William K., “*Digital Image Processing*”, John Wiley & Sons Inc., New York, 1991

[44] Prenter, P.M, “*Splines and Variational Methods*”, John Wiley and Sons, New York, 1989.

[45] Press, William H., Saul A. Teukolsky, William T. Vetterling and Brian P. Flannary, “*Numerical Recipes in C: The Art of Scientific Computing*”, Second Edition, Cambridge University Press, 1992.

[46] Poulin, Pierre and John Amanatides, “Shading and Shadowing with Linear Light Sources,” *Proceedings of Eurographics ‘90*, pp. 365-376, Elsevier Science Publishers B.V. (North-Holland) 1990.

[47] David F. Rogers and J. Alan Adams, “*Mathematical Elements for Computer Graphics,*” (Second Edition) McGraw-Hill Publishing Co., 1990.

[48] Rushmeier Holly, and Kenneth E. Torrance, “Extending the Radiosity Method to Include Specularly Reflecting and Translucent Materials,” *ACM Transactions on Graphics*, vol. 9, no. 1, pp. 1-27, January 1990.

[49] Salesin, David, Dani Lischinski, and Tony DeRose, “Reconstructing Illumination Functions with Selected Discontinuities,” *3rd Eurographics Workshop on Rendering*, Bristol, UK May 1992.

[50] Salmon, G, “*Analytic Geometry of Three Dimensions*”, Cambridge, England, Metcalfe, 1874.

[51] Schumacher Dale, and Chuck Seum, Public Domain Polygon Convexifier Program, posted to the USENET from dal@syntel.mn.org.

[52] Shroeder, Peter, “Numerical Integration for Radiosity in the Presence of Singularities”, *4th Eurographics Workshop on Rendering*, Paris, France June 1993.

[53] Shirley, Peter, “A Ray Tracing Method for Illumination Calculation in Diffuse-Specular Scenes,” *Proceedings of Graphics Interface ’90*, pp. 205-210, 1990.

[54] Siegel, Robert and John R. Howell, “*Thermal Radiation Heat Transfer*”, McGraw-Hill Book Co, 1972.

[55] Sillion Francois, and Claude Puech, “A General Two-Pass Method Integrating Specular and Diffuse Reflection,” *ACM Computer Graphics (SIGGRAPH ’89 Proceedings)*, vol. 23, no. 3, August 1989.

[56] Stewart, A. James and Sherif Ghali, “An Output Sensitive Algorithm for the Computation of Shadow Boundaries,” *Fifth Canadian Conference on Computational Geometry*, August 1993.

[57] Smits, Brian E., James R. Arvo, and David H. Salesin, “An Importance Driven Radiosity Algorithm,” *Computer Graphics (SIGGRAPH ‘92 Proceedings)*, vol. 26, no. 2, pp. 283-292, July 1992.

[58] Tanaka, Toshimitsu and Tokiichiro Takahashi, “Shading with Area Light Sources,” *Proceedings of Eurographics ‘91*, Elsevier Science Publishers B.V. (North-Holland) 1991.

[59] Tampieri, Fillipo and Dani Lischinski, “The Constant Radiosity Assumption Syndrome,” *2nd Eurographics Workshop on Rendering*, Barcelona, Spain, May 1991.

[60] Tampieri Fillipo, “Discontinuity Meshing for Radiosity Image Synthesis,” *Ph.D. Thesis*, Department of Computer Science, Cornell University, Ithaca, New York, 1993.

[61] Teller Seth, and Pat Hanrahan, “Global Visibility Algorithms for Illumination Computations,” *ACM Computer Graphics (SIGGRAPH ’93 Proceedings)*, vol. 27, no. 4, August 1993.

[62] Teller Seth, “Computing the Antipenumbra of an Area Light Source,” *ACM Computer Graphics (SIGGRAPH ’92 Proceedings)*, vol. 26, no. 2, pp. 139-148, July 1992

[63] Teller Seth, “Visibility Computations in Densely Occluded Polyhedral Environments”, *Ph.D. thesis*, University of California at Berkeley, 1992.

[64] Troutman, Roy and Nelson Max, “Radiosity Algorithms Using Higher-order Finite Elements,” *ACM Computer Graphics (SIGGRAPH ‘93 Proceedings)*, August 1993.

[65] Vedel, Christophe, “Computing Illumination from Area Light Source by Approximate Contour Integration,” *Proceedings of Graphics Interface ’93*, Toronto, Ontario, Canada, May 1993.

[66] Vedel, Christophe, “Improved Storage and Reconstruction of Light Intensities on Surfaces,” *3rd Eurographics Workshop on Rendering*, Bristol, UK May 1992.

[67] Wallace, John R., Kells A. Elmquist, and Eric A. Haines, "A Ray-Tracing Algorithm for Progressive Radiosity," *ACM Computer Graphics (SIGGRAPH '89 Proceedings)*, vol. 23, no. 3, July 1989.

[68] Wallace, John R., Michael F. Cohen and Donald P. Greenberg, "A Two Pass Solution to the Rendering Equation: A Synthesis of Ray Tracing and Radiosity Methods," *ACM Computer Graphics (SIGGRAPH '87 Proceedings)*, vol. 21, no. 4, July 1987.

[69] Ward, Gregory J., Francis M. Rubinstein and Robert D. Clear, "A Ray Tracing Solution for Diffuse Interreflection," *ACM Computer Graphics (SIGGRAPH '88 Proceedings)*, vol. 22, no. 4, August 1988.

[70] Ward, Gregory J. and Francis M. Rubinstein, "A New Technique for Computer Simulation of Illuminated Spaces," *Journal of the Illuminating Engineering Society*, vol. 17, no. 1, pp. 80-91, Winter 1988.

[71] Weiler, K. and P. Atherton, "Hidden Surface Removal Using Polygon Area Sorting," *ACM Computer Graphics (SIGGRAPH '77 Proceedings)*, vol. 4, no. 3, pp. 214-222, July 1977.

[72] Zatz, H. R., "Galerkin Radiosity: A Higher Order Solution Method for Global Illumination," *ACM Computer Graphics (SIGGRAPH '93 Proceedings),* August 1993.

Appendix A

A.1 Proof of Lemma 3-1

Lemma 1: *There exists only one value t_{min} of t on the line $L(t)$ such that the resulting sphere $S(t_{min})$ is tangential to the plane $z = 0$ at point P_{max}, and thus has only one point in common with the plane.*

Proof: From the discussion in Section 3.1.1 it can be seen that the equation of the sphere $S(t)$ is given as follows:

$$S(t) \equiv (x - (c_x + tn_x))^2 + (y - (c_y + tn_y))^2 + (z - (c_z + tn_z))^2 = t^2 + a^2 \tag{A-1}$$

If $z = 0$ is substituted into the equation of the receiver plane in Eq. (A-1), we get the following expression:

$$(x - (c_x + tn_x))^2 + (y - (c_y + tn_y))^2 = -(c_z + tn_z)^2 + (t^2 + a^2) \tag{A-2}$$

From Eq. (A-2) it can be seen that the solution is always a circle embedded in the $z = 0$ plane, and that the right hand side of the equation is the radius of this circle. If this right hand side is set equal to zero, and the roots of the equation thus formulated are found, exactly those values of t for which the sphere $S(t)$ is tangential to the plane $z = 0$ are given. If it can be determined that the equation has only one root of interest, then the lemma proof is complete. Setting the right hand equation of Eq. (A-2) to zero, gives the following quadratic and corresponding roots:

$$(n_z^2 + 1)t^2 + (2c_z n_z)t + (c_z^2 + a^2) = 0 \tag{A-3}$$

$$r_{1,2} = \frac{c_z n_z \pm (c_z^2 + n_z^2 a^2 - a^2)^{1/2}}{1 - n_z^2} \tag{A-4}$$

What do these two roots correspond to? If the x-axis is aligned with the centre of the sphere and the line $L(t)$, such that $C = (0, 0, c_z)$ and the unit normal $N = (n_x, 0, n_z)$ the result is the configuration shown in Figure A-1(b) in the z-x plane. From the geometry, the extension of the line AB, corresponding to the disc, meets the x-axis at $x_1 = (n_z/n_x) c_z$. For the configuration to have physical meaning, the tangential point x_{max} must be smaller than x_1, i.e.

$$x_{max} < x_1 \tag{A-5}$$

since if it is larger it is behind the source and receives no light. In addition, it is required that the source does not intersect the receiving plane. This condition is expressed by the following:

$$c_z > n_x a \tag{A-6}$$

Of the two roots in Eq. (A-4), only r_2 satisfies the property of Eq. (A-5). The two roots correspond to two x-values, x_{r_1} and x_{r_2}, that are the x coordinates of the centres of the circles (and corresponding spheres), as shown in Figure 1(b). It therefore suffices to show:

$$x_{r_1} > x_1 \tag{A-7}$$

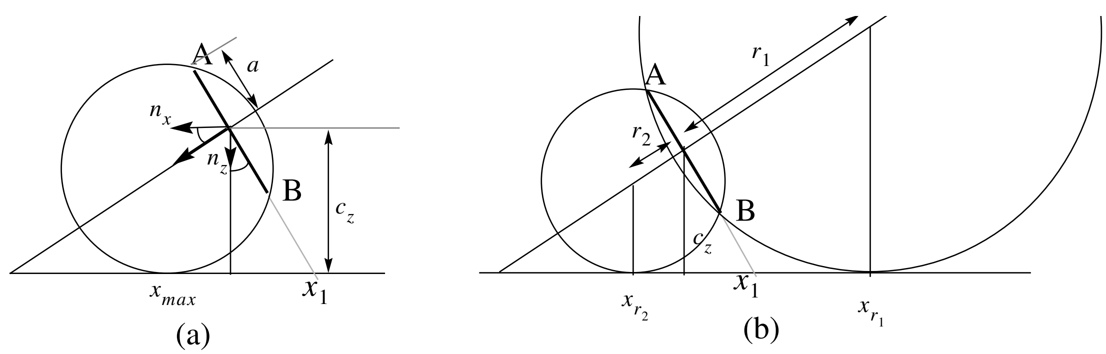


Figure A-1. Showing the Single Tangential Sphere

Eq. (A-7) is written:

$$x_r^1 > x_1 \Leftrightarrow r_1 n_x > (n_z/n_x) c_z \Leftrightarrow \frac{c_z n_z + (c_z^2 + n_z^2 a^2 - a^2)^{1/2}}{1 - n_z^2} n_x > (n_z/n_x) c_z \Leftrightarrow \tag{A-8}$$

$$(c_z^2 + n_z^2 a^2 - a^2)^{1/2} > c_z n_z \left(\frac{(1 - n_z^2) - n_x^2}{n_x^2} \right) = 0 \tag{A-9}$$

since $n_z^2 + n_x^2 = 1$. But from Eq. (A-6) we have:

$$c_z^2 + n_z^2 a^2 - a^2 = c_z^2 + a^2 (n_z^2 - 1) > n_x^2 a^2 + a^2 (n_z^2 - 1) = a^2 (n_x^2 + n_z^2 - 1) = 0 \Rightarrow \tag{A-10}$$

$$(c_z^2 + n_z^2 a^2 - a^2)^{1/2} > 0 \tag{A-11}$$

and therefore Eq. (A-9) is true. □

A.2 Unimodality for Rectangle and Parallel Receiver

For the case of a rectangular source directly opposing a receiver plane parallel to the plane of the source, the equation of illumination is much simplified[1]. If the centre of the coordinate system is in the centre of the rectangular source, and the source extents are described by the variables x_1, y_1 that vary $-a < x_1 < a$ and $-b < y_1 < b$, then the function $L(x, y)$ is expressed as follows:

$$L(x, y) = \int_{-a}^{a} \int_{-b}^{b} \frac{dy_1 dx_1}{((x - x_1)^2 + (y - y_1)^2 + d^2)^2} = \int_{x-a}^{x+a} \int_{y-b}^{y+b} \frac{dy_1 dx_1}{(x_1^2 + y_1^2 + d^2)^2} \tag{A-12}$$

Since $\frac{d}{dx_1}(x - x_1) = dx_1$ and similarly for y_1, we can directly apply the antidervative to find $\frac{\partial}{\partial x} L(x, y)$ and $\frac{\partial}{\partial y} L(x, y)$.

$$\frac{\partial}{\partial x} L(x, y) = \int_{y-b}^{y+b} \left\{ \frac{1}{((x + a)^2 + y_1^2 + d^2)^2} - \frac{1}{((x - a)^2 + y_1^2 + d^2)^2} \right\} dy_1 \tag{A-13}$$

We can now examine the integrand in Eq. (13). Call $f(u) = (u^2 + y_1^2 + d^2)^2$.Then the integrand in Eq. (13) is:

$$\frac{1}{f(x + a)} - \frac{1}{f(x - a)} = \frac{f(x - a) - f(x + a)}{f(x + a) f(x - a)} = \frac{-2xa}{f(x + a) f(x - a)} \tag{A-14}$$

But the denominator is always positive, and numerator has the opposite sign of *x*. But then the integral also has the opposite sign of *x*. Similarly for the partial in *y*. Define $r = \sqrt{x^2 + y^2}$. Then:

1. Much of the following derivation is due to personal communication with Arthuer L. Rubin.

$$L(x, y) = \frac{1}{r^2} \left(x \frac{\partial}{\partial x} L(x, y) + y \frac{\partial}{\partial y} L(x, y) \right) \quad \text{(A-15)}$$

is always negative. Consequently $L(x, y)$ decreases as a function of the distance from the maximum, which is a condition more powerful than that required for our definition of unimodality.□

Appendix B

B.1 Coefficients of the EEE Quadric Surface

This derivation is from Gigus, Canny and Seidel [26], and we have expanded the equations to give the final coefficient expressions. Consider three edges $\{e_1, e_2, e_3\}$ shown in Figure B-2. Let the endpoints of each edge be points $\mathbf{a}_i$ and $\mathbf{b}_i$. Then consider a point $\mathbf{p}$ that is on the quadric surface. Call P_i the plane defined by edge e_i and point $\mathbf{p}$. Define $\mathbf{d}_i$ to be the direction vector defines

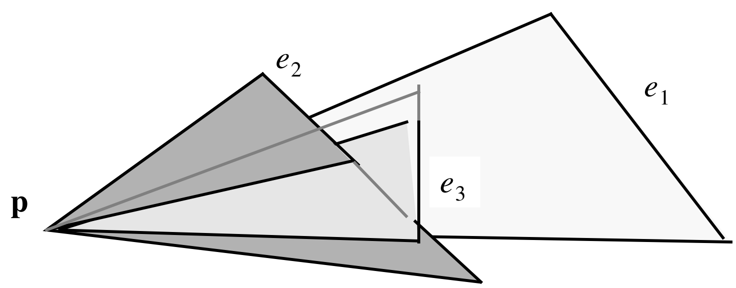


Figure B-2. Three Edges of a EEE Surface

as $\mathbf{d}_i = \mathbf{b}_i - \mathbf{a}_i$. Then the normal to the plane P_i is:

$$N_i = (\mathbf{p} - \mathbf{a}_i) \times \mathbf{d}_i. \tag{B-16}$$

Then the direction along the line of intersection of two planes P_i and P_j is:

$$P_{i,j}(\mathbf{p}) = N_i \times N_j. \tag{B-17}$$

This line is the unique line l_c that passes through the point **p** and the edges e_i and e_j. Thus, if we require a line that cuts all three edges and passes through **p** we get the equation:

$$\langle P_{1,2}(\mathbf{p}), N_3 \rangle = 0. \qquad \text{(B-18)}$$

Equation Eq. (B-18) is a quadric equation in the coordinates of **p**, and is used to determine the coefficients of Eq. (4-1), shown below as Eq. (B-19).

$$Ax^2 + By^2 + Cz^2 + Dyz + Exz + Fxy + Gx + Hy + Iz + J = 0 \qquad \text{(B-19)}$$

Let:

$$\mathbf{a}_i = (a_{ix}, a_{iy}, a_{iz}) \qquad \mathbf{b}_i = (b_{ix}, b_{iy}, b_{iz}) \qquad \mathbf{d}_i = (d_{ix}, d_{iy}, d_{iz}) \qquad i = 1, 2, 3 \qquad \text{(B-20)}$$

Given the following subsitutions:

$$\begin{array}{lll} c_1 = a_{1x}d_{1z} - a_{1z}d_{1x} & c_2 = a_{2y}d_{2x} - a_{2x}d_{2y} & c_3 = a_{2x}d_{2z} - a_{2z}d_{2x} \\ c_4 = a_{1z}d_{1y} - a_{1y}d_{1z} & c_5 = a_{1y}d_{1x} - a_{1x}d_{1y} & c_6 = a_{2z}d_{2y} - a_{2y}d_{2z} \\ c_7 = a_{3y}d_{3x} - a_{3x}d_{3y} & c_8 = d_{1x}d_{2y} - d_{1y}d_{2x} & c_9 = a_{3z}d_{3y} - a_{3y}d_{3z} \\ c_{10} = a_{3x}d_{3z} - a_{3z}d_{3x} & c_{11} = d_{1y}d_{2z} - d_{1z}d_{2y} & c_{12} = d_{1z}d_{2x} - d_{1x}d_{2z} \end{array}, \qquad \text{(B-21)}$$

the coefficients of the quadric are determined from Eq. (B-18) and Eq. (B-21). The expression was given to a symbolic manipulation program (Maple), which subsequently collected the terms and gave the coefficients shown below.

$$A = -(d_{1y}c_6 - c_4d_{2y})d_{3z} + c_{11}c_9 + (d_{1z}c_6 - c_4d_{2z})d_{3y} \qquad \text{(B-22)}$$

$$B = -(d_{1z}c_3 - c_1d_{2z})d_{3x} + (d_{1x}c_3 - c_1d_{2x})d_{3z} + c_{12}c_{10} \qquad \text{(B-23)}$$

$$C = (d_{1y}c_2 - c_5d_{2y})d_{3x} - (d_{1x}c_2 - c_5d_{2x})d_{3y} + c_8c_7 \qquad \text{(B-24)}$$

$$\begin{aligned} D = {} & (d_{1x}c_2 - c_5d_{2x})d_{3z} + (c_5d_{2z} - d_{1x}c_6 + c_4d_{2x} - d_{1z}c_2)d_{3x} + c_8c_{10} \\ & - (-c_1d_{2x} + d_{1x}c_3)d_{3y} + c_{12}c_7 - (c_4d_{2x} - d_{1y}c_3 + c_1d_{2y} - d1_xc_6)d_{3x} \end{aligned} \qquad \text{(B-25)}$$

$$\begin{aligned} E = {} & (d_{1y}c_6 - c_4d_{2y})d_{3x} - (c_1d_{2y} - d_{1z}c_2 + c_5d_{2z} - d_{1y}c_3)d_{3y} + c_8c_9 \\ & - (-c_5d_{2y} + d_{1y}c_2)d_{3z} + (c_4d_{2x} - d_{1y}c_3 + c_1d_{2y} - d_{1x}c_6)d_{3y} + c_{11}c_7 \end{aligned} \qquad \text{(B-26)}$$

$$\begin{aligned} F = {} & (c_1d_{2y} - d_{1z}c_2 + c_5d_{2z} - d_{1y}c_3)d_{3z} + c_{12}c_9 + (d_{1z}c_3 - c_1d_{2z})d_{3y} + c_{11}c_{10} \\ & - (c_5d_{2z} - d_{1x}c_6 + c_4d_{2x} - d_{1z}c_2)d_{3z} - ((-c_4d_{2z} + d_{1z}c_6)d_{3x}) \end{aligned} \qquad \text{(B-27)}$$

$$\begin{aligned} G = {} & (c_1d_{2y} - d_{1z}c_2 + c_5d_{2z} - d_{1y}c_3)c_9 - (c_5c_6 - c_4c_2)d_{3z} + (-c_4d_{2z} + d_{1z}c_6)c_7 \\ & + (d_{1y}c_6 - c_4d_{2y})c_{10} + (c_4c_3 - c_1c_6)d_{3y} \end{aligned} \qquad \text{(B-28)}$$

$$H = (c_1c_2 - c_5c_3)\,d_{3z} + (c_5d_{2x} - d_{1x}c_6 + c_4d_{2x} - d_{1z}c_2)\,c_{10} + (-c_1d_{2x} + d_{1x}c_3)\,c_9$$
$$(d_{1z}c_3 - c_1d_{2x})\,c_7 - (c_4c_3 - c_1c_6)\,c_3 \tag{B-29}$$

$$I = -(c_1c_2 - c_5c_3)\,d_{3y} + (d_{1x}c_2 - c_5d_{2x})\,c_9 + (c_5c_6 - c_4c_2)\,d_{3x} + (-c_5d_{2y} + d_{1y}c_2)\,c_{10}$$
$$(c_4d_{2x} - d_{1y}c_3 + c_1d_{2y} - d_{1x}c_6)\,c_7 \tag{B-30}$$

$$J = (c_1c_2 - c_5c_3)\,c_9 + (c_4c_3 - c_1c_6)\,c_7 + (c_5c_6 - c_4c_2)\,c_{10} \tag{B-31}$$

Appendix C
Colour Images

Colour Image 1. Frames 2 & 13 of Test Suite LS Translation (Shaded Structured Solution)

Colour Image 2. Frames 2 & 13 of Test Suite LS Translation (Shaded Unif. Quad. Solution)

Colour Image 3. Error Frames 2 and 13 of Test Suite LS Translation (Struct. Solution)

Colour Image 4. Error Frames 2 and 13 of Test Suite LS Translation (Unif. Quad. Solution)

Colour Image 5. (a) One Desk and (b) Two Desks and Bookcase Analytic Image

Colour Image 6. (a) 1 Desk, 1 Chair and (b) 2 Desks, 2 Chairs & Bookcase Analytic Image

Colour Image 7. (a) Structured Solution and (b) Error Image

Colour Image 8. (a) Reduced Mesh Solution and (b) Error Image

Colour Image 9. (a) Degree Selection Solution and (b) Error Image

Colour Image 10. (a) Desk and Chair Structures Solution and (b) Error Image